CS 200

Lecture 09
FileMaker vs SQL & Reports
Miscellaneous Notes

Abbreviations

• aka also known as
• DBMS DataBase Management System
• *mutatis mutandis* with the necessary changes having been made
Administrivia

Please read and highlight BEFORE lab

• Assignment 9
• This week’s lecture slides

Reading

• Database Design (on learn.uwaterloo.ca)
  Adapted from Access Database Design and Programming by Steven Roman

Today

• SQL vs FileMaker
• Reports in FileMaker
• Read and Reason Pearl

Please ask questions!
Databases—FileMaker

This is your second pass at FileMaker

• The first time around, you got a rough idea of what it does
• This time round we’ll
  fill in some important details
  evaluate its power & flexibility

Our strategy

• SQL as an application model for FileMaker
• so we’ll “compare and contrast...” the two

Recall the Albums and Songs database

Arrows go FROM a foreign key TO a corresponding primary key (they DO NOT have to go between fields with the same name!)
Simple One-Table Queries (1)

In SQL:
- `select` Title, Artist, Price `from` Albums

In FileMaker:
- make a new layout
- choose, position, label and format fields
- enter browse mode

In each vs the other ...
- what’s different?
- what’s better?
- what’s worse?
- what’s easier?
- what’s harder?
- what can’t be done?
Simple One-Table Queries (2)

In SQL:
• `select` Title, Artist `from` Albums `where` Price < 10.00

In FileMaker:
• make another new layout
• choose, position, label, & format fields
• enter find mode & type the condition ("query by example")
• then perform the find

In each vs the other ...
• what’s different? better? worse? easier? harder? can’t be done?
One-Table Queries (and)

How about...

```sql
select Title, Artist from Albums
   where (Price < 14.00) and (Artist = 'Mozart') ?
```

Hmm ... what’s the “Omit” check box do? (Read and Remember...)
One-Table Queries (or)

How about

```
select Title, Artist from Albums
where (Price < 10.00) or (Artist = 'Mozart') ?
```

FileMaker calls each line of a Find a "request"

In SQL I didn’t have to display Price to use it in a query; must I in FileMaker?
One-Table Queries (Using a Field Twice – or)

How about

```sql
select Title, Artist from Albums
where (Price < 10.00) or (Price > 15) ?
```

Are there any SQL queries that FileMaker can’t do?

How about \((a < b) \text{ or } [(c < d) \text{ and } (e < f)]\)?

Roughly speaking, there’s a theorem that you can express all boolean expressions with all or’s or with all and’s

\(\text{In the above example, } (a < b) \text{ or } [(c < d) \text{ and } (e < f)] = (a < b) \text{ or } ((c \geq d) \text{ or } (e \geq f))\)

But what about sub-selects??? Recall

```sql
select ... from ... where ( ... ( select ... from ... where ... ) ... )
```
Suppose I want to use Price twice in a single “request”?

```sql
select Title, Artist from Albums
where (Price > 10.00) and (Price < 15)
order by Price ?
```
Two Table Queries/Joins in SQL

Select  Albums.Title as "Album Title", Songs.Title as "Song Title"
From    Albums, Songs
Where   Albums.Album_ID  =  Songs.Album_ID
Order By Albums.Album_ID, Song_ID
Two-Table Queries/Joins in FileMaker (From the Songs Table)

From the Songs Table

... where Songs.Album ID = Albums.Album ID

Songs is the “master table” in FileMaker-speak

This is a “many-to-one” relationship from Song to Album in FileMaker-speak
The Result (Made to Look Like SQL Output)

Of course, you’d almost never actually DO this in FileMaker, since you can make much more readable and attractive layouts.
Two-Table Queries/Joins in FileMaker (From the Albums Table)

But from the Album Table’s point of view (it’s now the “master table”)

...where Albums.Album ID = Songs.Album ID

This is a “one-to-many” relationship from Albums to Songs

Could I create the preceding layout in the Albums table?
The Result — First Formatted to Look Like SQL

One to Many

<table>
<thead>
<tr>
<th>Album ID</th>
<th>Album Title</th>
<th>Artist</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1 Once Upon a Time</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>2 A Love for Life</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>3 Nice to Meet You</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>4 So Long My Friend</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>5 You Only Live Once</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>6 To the One Who Knows</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>7 Face in the Photograph</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>8 Fellisa</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>9 Desire</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>10 Aria</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>11 A Night to Remember</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>12 In the Mirror</td>
</tr>
</tbody>
</table>

But notice the use of a portal.

A layout that’s more natural for FileMaker, and easier to read.

<table>
<thead>
<tr>
<th>Song ID</th>
<th>Side</th>
<th>Track Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>Once Upon a Time</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>A Love for Life</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>Nice to Meet You</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>So Long My Friend</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>You Only Live Once</td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>To the One Who Knows</td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>Face in the Photograph</td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>Fellisa</td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>Desire</td>
</tr>
<tr>
<td>10</td>
<td>10</td>
<td>Aria</td>
</tr>
<tr>
<td>11</td>
<td>11</td>
<td>A Night to Remember</td>
</tr>
<tr>
<td>12</td>
<td>12</td>
<td>In the Mirror</td>
</tr>
</tbody>
</table>

Album fields. Song fields.
Multi-Key (Two-Table) Queries/Joins

From last week’s University sample database

<table>
<thead>
<tr>
<th>Mark</th>
<th>Assignment</th>
</tr>
</thead>
<tbody>
<tr>
<td>StudNum</td>
<td>Course</td>
</tr>
<tr>
<td>Course</td>
<td>AssignNum</td>
</tr>
<tr>
<td>AssignNum</td>
<td>Weighting</td>
</tr>
<tr>
<td>Mark</td>
<td>Description</td>
</tr>
<tr>
<td></td>
<td>Date Due</td>
</tr>
</tbody>
</table>

List of assignment due dates for student 99001122

```sql
select M.StudNum, M.Course, M.AssignNum, Date Due
from Mark M, Assignment A
where ( M.StudNum = 99001122 )
and ( M.Course = A.Course )
and ( M.AssignNum = A.AssignNum )
```

How is this done in FileMaker?

- grab a field in one table, drag it to a field in another or use buttons, selected fields, and popups

Notice how the structure of this dialog reflects the natural syntax of a where.
Are there 2-table SQL queries that can’t be done in FileMaker?

Can these?

```sql
select * from Albums, Songs
```

```sql
select * from Albums, Songs where Album_ID < Songs_ID
```

How about using a table twice in an SQL join (eg “select ... from Song as S1, Song as S2”)?

In Filemaker, create “copies” of a table in FileMaker’s relationships dialog when you would list the table > once in the corresponding SQL where clause.

Are there 2-table FileMaker queries that can’t be done in SQL?
FileMaker Terminology (1)

**FileMaker File or Document**
- a collection of one or more database tables, including field definitions
  - + layouts (aka “views”)
  - + scripts

**Master Table**
- the file that accesses & displays data from another file
  - one or more records in a “related file” that is/are identified “via a relationship”

**Related Table**
- a file containing related data you want to access in the master file

**Match Fields**
- a field in the master file and a field in the related file that contain values you want to use to find matching records
  - aka “link fields”
FileMaker Terminology (2)

Master Record
A record in the master table for which you wish to find matching records in a related table.

Related Record
A record in the related file whose link field (according to the relationship used) contains a value equal to that of the link field of the master record.

Related Field
A field in a related record.
May be placed directly on a layout of the master table, or in the first row of a portal (also in a layout of the master table).

Portal
A layout object in the master file in which you place related fields.
Use a portal when the relation involved is one-to-many.
(You don’t need it when the relation involved is many to one.)
From Last Week’s Lecture

List the rooms in which students have lectures

```
select S.IDN, S.Name, R.Course, C.Room
from Students S, Register R, Courses C
where (S.IDN = R.IDN)
and (R.Course = C.Name)
order by S.Name, C.Course
```

Here’s what we want for each student…

with Students as the master table

with Courses as the master table (+ a query for Aaron)
Hmm ... in a Students layout, how do you access Courses

Thus

you can relate (link) A to B, relate B to C,
then on a layout in A use fields from C as well as B

If you position the field in a portal, FileMaker will show all the related values

If you position the field outside a portal, FileMaker will show you the first related value it finds

which is just what you want if the relationship is many-to-one
but “wrong” if the relationship is one-to-many

however, FileMaker doesn’t care — it just does what you tell it to...
Study Questions

What about 4-table joins? 5-table joins? etc

Which is more powerful: SQL or FileMaker?

Which is easier to use: SQL or FileMaker?

If SQL is the definition of a (truly) “relational database” ...

  is FileMaker a relational database?
  if it isn’t, why do so many people buy FileMaker?
Reports in FileMaker

Reports are something SQL barely does

• a possible reason for preferring FileMaker
• (though note that you can purchase “report generators” that hook to SQL DBMSs)

A report is just a list of records

• ie field values in each of those records
• and possibly in linked records from linked tables

You lay out how you want those fields arranged

• FileMaker prints that layout repeatedly, once for each record in the “found set”
• usually you don’t want to report on every record, so you do a query first; what’s printed is data from records in the “found set”

But usually you’d like to

• order the records
• and/or to group them
• and print summary information about each group, eg the number of records in each group, the average / total / etc of some field
Groups

And usually some of the information is the same within a group

- it would be nice to only print it once
- (e.g., the Album Title is the same for all the songs on an album)

So...

We’d like a way to

“group” records and

“do something” once for each group

What’s a “group” in FileMaker?

- sort on some field’s value
- each sequence of records with the same field value is a group

What does it mean to “do something” for each group?

print once

the fields that don’t change and/or

summary (“aggregate”) information

between the records for one group
and the records for the next group
## Organization of a Report Layout

<table>
<thead>
<tr>
<th>Event</th>
<th>Section</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>At the top of every page</td>
<td>Header</td>
<td>printed once on each page</td>
</tr>
<tr>
<td>Precedes the first leading subsummary</td>
<td>Leading Grand Summary</td>
<td>printed once</td>
</tr>
<tr>
<td>Precedes every change in the sorted field’s value</td>
<td>Leading SubSummary</td>
<td></td>
</tr>
<tr>
<td>Repeated for Every Record</td>
<td>BODY</td>
<td>printed once for each group</td>
</tr>
<tr>
<td>Follows every change the sorted field’s value</td>
<td>Trailing SubSummary</td>
<td></td>
</tr>
<tr>
<td>Follows the last trailing subsummary</td>
<td>Trailing Grand Summary</td>
<td>printed once</td>
</tr>
<tr>
<td>At the bottom of every page</td>
<td>Footer</td>
<td>printed once on each page</td>
</tr>
</tbody>
</table>
In FileMaker . . .

- Each sub-summary part is associated with a “break field”
  whose distinct values define groups
  But using the sub-summary does not cause a sort to be done on the break field
  so as to create the groups!
  — you must do that yourself, separately

- Browse mode only shows sub-summaries if it’s sorted by the break field
  For that you must be in Preview mode (yuch)
  And you must have sorted by the relevant break field

Remember “Group By” in SQL?

- it told SQL what field(s) to group records by
  when using an aggregate function [eg count() and sum()]

- it’s the same idea in FileMaker
  except that sorting serves as both Order By & Group By

- SQL aggregate functions = FileMaker summary fields
Example of a Report Layout

The first album on this page is: ::Title

Category ::Category Medium ::Medium Copyright ::Copy Running Time Album ::mins

Album contents as of: {{CurrentDate}}
The Resulting Report

### God Shuffled His Feet

**Category**: Soft Rock  
**Medium**: CD  
*Copyright 1993  Running Time 50.8833 mins*

<table>
<thead>
<tr>
<th>Track No</th>
<th>Time</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5:10</td>
<td>God Shuffled His Feet</td>
</tr>
<tr>
<td>2</td>
<td>3:56</td>
<td>Afternoons and Coffeespoons</td>
</tr>
<tr>
<td>3</td>
<td>3:55</td>
<td>MMM MMM MMM MMM MMM</td>
</tr>
<tr>
<td>4</td>
<td>3:40</td>
<td>In the Days of the Caveman</td>
</tr>
<tr>
<td>5</td>
<td>5:07</td>
<td>Livin’ on the Edge</td>
</tr>
<tr>
<td>6</td>
<td>3:49</td>
<td>Swimming in Your Ocean</td>
</tr>
<tr>
<td>7</td>
<td>3:06</td>
<td>Here I Stand Before Me</td>
</tr>
<tr>
<td>8</td>
<td>4:52</td>
<td>I Think I’ll Disappear Now</td>
</tr>
<tr>
<td>9</td>
<td>3:42</td>
<td>How Does a Duck Know?</td>
</tr>
<tr>
<td>10</td>
<td>3:43</td>
<td>When I Go Out with Artists</td>
</tr>
<tr>
<td>11</td>
<td>3:47</td>
<td>The Psychic</td>
</tr>
<tr>
<td>12</td>
<td>3:24</td>
<td>Two Knights and Maidens</td>
</tr>
<tr>
<td>13</td>
<td>3:42</td>
<td>Untitled</td>
</tr>
</tbody>
</table>

**Live: Right here, right now**

**Category**: Hard Rock  
**Medium**: CD  
*Copyright 1993  Running Time 141.266 mins*

<table>
<thead>
<tr>
<th>Track No</th>
<th>Time</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>5:26</td>
<td>Poundcake</td>
</tr>
<tr>
<td>2</td>
<td>4:52</td>
<td>Judgement Day</td>
</tr>
<tr>
<td>3</td>
<td>2:22</td>
<td>When it’s Love</td>
</tr>
<tr>
<td>4</td>
<td>5:06</td>
<td>Spanked</td>
</tr>
<tr>
<td>5</td>
<td>4:37</td>
<td>Ain’t Talkin’ ’Bout Love</td>
</tr>
<tr>
<td>6</td>
<td>6:21</td>
<td>In ‘N’ Out</td>
</tr>
<tr>
<td>7</td>
<td>4:49</td>
<td>Dreams</td>
</tr>
<tr>
<td>8</td>
<td>4:50</td>
<td>Man on a Mission</td>
</tr>
<tr>
<td>9</td>
<td>5:14</td>
<td>Ultra Base</td>
</tr>
<tr>
<td>10</td>
<td>9:38</td>
<td>Pleasure Dome/Drum solo</td>
</tr>
<tr>
<td>11</td>
<td>6:39</td>
<td>Panama</td>
</tr>
<tr>
<td>12</td>
<td>5:14</td>
<td>Love Walks In</td>
</tr>
<tr>
<td>13</td>
<td>5:21</td>
<td>Runaround</td>
</tr>
<tr>
<td>2</td>
<td>1:18</td>
<td>Right Now</td>
</tr>
</tbody>
</table>

*Album contents as of: Wednesday, 11 November 2013*
Multi-Level Reports – Subgroups

Suppose you want to break a group into subgroups

and print some summary information for each subgroup?

eg separately for side 1 and side 2 of an album

It works just as you’d expect

sort first by the outer group, then by the inner

put a (another) sub-summary part between the

outer sub-summary part & the body part, and

set its break field to be the second sort key
The Resulting Two-Level Report (by Album and Side)

At the top of every page | Header
---|---
Precedes the first leading subsummary | Leading Grand Summary
Precedes every change in the primary sort field's value | Leading Outer SubSummary
Precedes every change in the secondary sort field's value | Leading Inner SubSummary
Repeated for every record | BODY
Follows every change in the secondary sort field's value | Trailing Inner SubSummary
Follows every change in the primary sort field's value | Trailing Outer SubSummary
Follows the last trailing subsummary | Trailing Grand Summary
At the bottom of every page | Footer

---

### God Shuffled His Feet

<table>
<thead>
<tr>
<th>Category</th>
<th>Soft Rock</th>
<th>Medium CD</th>
<th>Copyright 1993</th>
<th>Total Running Time</th>
<th>50.88 mins</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Side</th>
<th>Track</th>
<th>Mins</th>
<th>Secs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Minutes this side:</td>
<td>50.88</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>10</td>
<td>God Shuffled His Feet</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>56</td>
<td>Afternoons and Coffeespoons</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>55</td>
<td>MMM MMM MMM MMM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>40</td>
<td>In the Days of the Caveman</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>07</td>
<td>Livin' on the Edge</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>49</td>
<td>Swimming in Your Ocean</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>06</td>
<td>Here I Stand Before Me</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>52</td>
<td>I Think I'll Disappear Now</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>42</td>
<td>How Does a Duck Know?</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>43</td>
<td>When I Go Out with Artists</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>47</td>
<td>The Psychic</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>24</td>
<td>Two Knights and Maidens</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>42</td>
<td>Untitled</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### Live: Right here, right now

<table>
<thead>
<tr>
<th>Category</th>
<th>Hard Rock</th>
<th>Medium CD</th>
<th>Copyright 1993</th>
<th>Total Running Time</th>
<th>? mins</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Side</th>
<th>Track</th>
<th>Mins</th>
<th>Secs</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Minutes this side:</td>
<td>73.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>28</td>
<td>Poundcake</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>52</td>
<td>Judgement Day</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>22</td>
<td>When it's Love</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>08</td>
<td>Spanked</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>37</td>
<td>Ain't Talkin' 'Bout Love</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>21</td>
<td>In 'N Out</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>49</td>
<td>Dreams</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>50</td>
<td>Man on a Mission</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>14</td>
<td>Ultra Bass</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>38</td>
<td>Pleasure Dome/Drum solo</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>39</td>
<td>Panama</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>14</td>
<td>Love Walks In</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>21</td>
<td>Runaround</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>2</th>
<th>Minutes this side:</th>
<th>67.71</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>13</td>
<td>Right Now</td>
</tr>
<tr>
<td>4</td>
<td>58</td>
<td>One Way to Rock</td>
</tr>
<tr>
<td>5</td>
<td>22</td>
<td>Why Can't This Be Love</td>
</tr>
<tr>
<td>5</td>
<td>39</td>
<td>Give to Live</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>Finish What Ya Started</td>
</tr>
<tr>
<td>5</td>
<td>00</td>
<td>Best of Both Worlds</td>
</tr>
<tr>
<td>7</td>
<td>37</td>
<td>316</td>
</tr>
<tr>
<td>8</td>
<td>58</td>
<td>You Really Got Me/Cabo Wabo</td>
</tr>
<tr>
<td>9</td>
<td>41</td>
<td>Won't Get Fooled Again</td>
</tr>
<tr>
<td>10</td>
<td>26</td>
<td>Jump</td>
</tr>
<tr>
<td>11</td>
<td>59</td>
<td>Top of the World</td>
</tr>
</tbody>
</table>

---

CS 200 Spring 2017
Reports — Some Details

Use the Format menu to control the appearance of data values

If a field appears in a Header

the value for the first record on the page is used

similarly for the footer, mutatis mutandis

Use the Define Fields dialog box

to create a summary field (SQL aggregate function)
that you place in a grand summary or sub-summary part
CHECK( ... ) in SQL

You can’t mistype something that’s entered for you!

About “Looked-up values”

- “copy the contents of a related field when the match field is initialized or when I say so (Records > Relookup)”
- these were FM’s first try at a relational database
- usually not what you want
An Important Principle:

It’s easier to catch & correct errors when data are entered than it is to find and correct them later.
Indexing in FileMaker

Remember SQL’s Create Index command? Here’s how FileMaker does it

**FileMaker will automatically index fields**

- to implement “Unique” or “Existing” data validation on the matched field in a related table when the field is used to create a value list provided you haven’t manually forced indexing off

**Otherwise, by default:**

- “Automatically turn indexing on if needed”

**In FileMaker**

- it’s unlikely you’ll ever need to manually turn indexing on but you may want to turn it off to save file space because FileMaker never will!
- Other databases make up their own rules

**And while we’re here**
Formats in FileMaker

Note especially the difference between

• how a data value appears on a form
  eg 3.14
• and what’s stored in the database
  eg 3.14159265358979323846264338327950288419716939937510582097494459230781
  7093844609550582

& note the menu items that control the appearance of each data type
The Inspector

The Inspector is a tool used to configure the appearance and behavior of elements within a FileMaker database. It allows users to adjust settings such as position, size, alignment, and formatting. The Inspector is divided into several sections, including:

- **Position**: Allows you to adjust the element's position on the screen.
- **Appearance**: Enables customization of the element's appearance, including fill, line, and text baseline settings.
- **Data**: Provides options for data formatting and tab positions.
- **Autosizing**: Helps in adjusting the size of the element automatically.

The Inspector is particularly useful for fine-tuning the design and user experience of database layouts.
Define Field Value Lists vs Format Field Value Lists

What’s the difference?
Data Validation (3) — Pop-up Menus / Drop-down Lists

And there are radio buttons and checkboxes...

which also allow users to choose from a value list
but display the choice differently.
Generalizing

If I select a field on a layout

• FileMaker grays out inappropriate Format menu items
• selecting one of the active menu items
  lets me change the appearance of the selected field

What happens if nothing is selected & I select a Format menu item? what does it mean?

Why is this an important question?
Referential Integrity

For every foreign key

- there is a record in the referenced table having that primary key
- eg for every Album_ID value appearing in a Song record there is an Album with that Album_ID.

A — “Cascading deletes”

- if you delete an album, the related songs are also automatically deleted

B — auto-creation of related song records

- just by typing into related fields on a layout
  - the foreign key is automatically set, too
  - be smart: auto-enter the song’s pKey

C — like B, but from Songs to Albums

D — like A, but from Songs to Albums

- would not be smart here ...
- what would happen if you check it, & later delete a song?
Things That Might Confuse You in FileMaker

You can define as many layouts at you want

• these are also called “views” of the data (SQL-speak)
• give them meaningful names!!!

You don’t have to include all fields on all layouts

• defining a new field doesn’t cause it to appear on all layouts
• by default, a new field is placed on the “current layout,” although there’s a preference to prevent that

In FileMaker

• data entry options are associated with a field’s definition
• the appearance of a field is specified on the layout, and can vary
• other databases may choose different conventions

You can only see sub-summaries

• when in Preview Mode or Browse Mode
  and if you have sorted by the relevant break field(s)
• doing both — and remembering to do both — is a pain

(Incidentally, FileMaker allows you to userid-/password-protect a table, layouts, etc, and to access a database over the internet (there’s a more expensive version tuned for this.)
Summing Up

SQL joins vs FileMaker joins ("relationships")

• FileMaker is now built on top of an SQL engine
• but does FileMaker’s GUI provide a way to generate and display all possible SQL selects?

FileMaker features typical of GUI-flavoured DBMS’s

• forms with
  formatting of displayed values
  data validation
  input widgets (radio buttons, check boxes, pop-ups, etc)
  point-and-click-to-open (next week)
  query by example

• reports and
  formatting of displayed values
  sorting
  summaries, sub-summaries, aggregate functions

Why might someone prefer SQL?

Why might someone prefer FileMaker?
The Read and Reason Pearl

When something goes wrong

• take the time to read the error message(s) carefully
• list all the symptoms as precisely as you can
• think about what this info says and implies

A good general strategy is to

• make a list of *possible* causes
• figure out how to test whether each *is* the cause
Example

What might be the problem here?

How do you test each possibility?

```
<html>
<head>
<title>Read 'n Reason</title>
</head>
<body bgcolor="#ffffff">
<p>This is how you embed graphic images in a web page:</p>
<p><center><img src="Images/Casper.jpeg"></center></p>
</body>
</html>
```
Read And Reason — The Methodology

Given a failure of some kind

• read what’s on the screen
• gather relevant data
• design a test for each
to determine if it’s the actual cause
• perform the tests

Often this is an iterative process

• you narrow down the set of possible causes at each iteration
• eg
  client, server, or network problem?
  system vs application
  application itself or 3rd party plugins
  etc