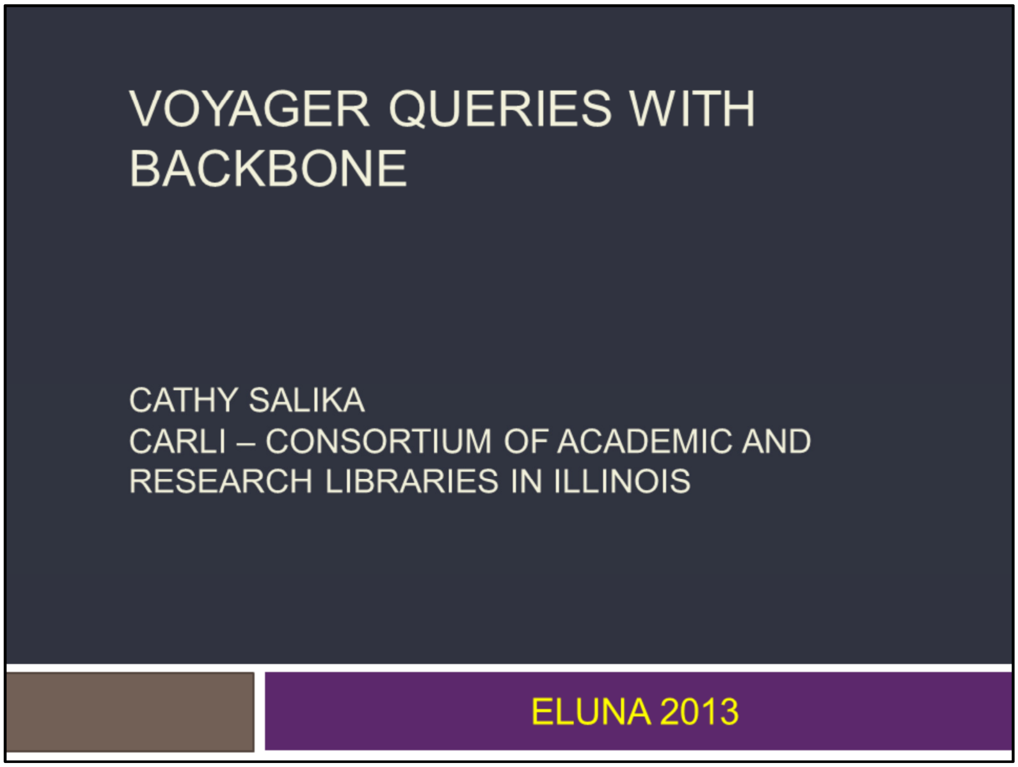


VOYAGER QUERIES WITH BACKBONE

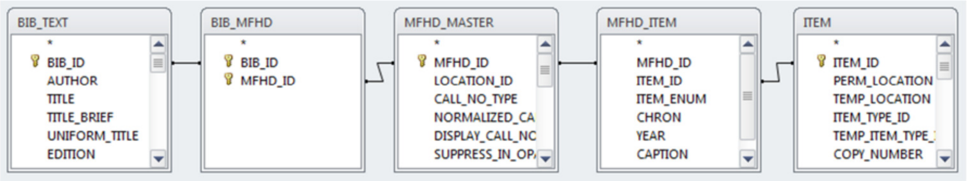
CATHY SALIKA
CARLI – CONSORTIUM OF ACADEMIC AND RESEARCH LIBRARIES IN ILLINOIS

ELUNA 2013

We’re going to focus on the top half of the query design pane
the way the tables are assembled.

Picking the right tables and linking them correctly makes it likely that your query results will be correct.

The Bib-to-Item Backbone



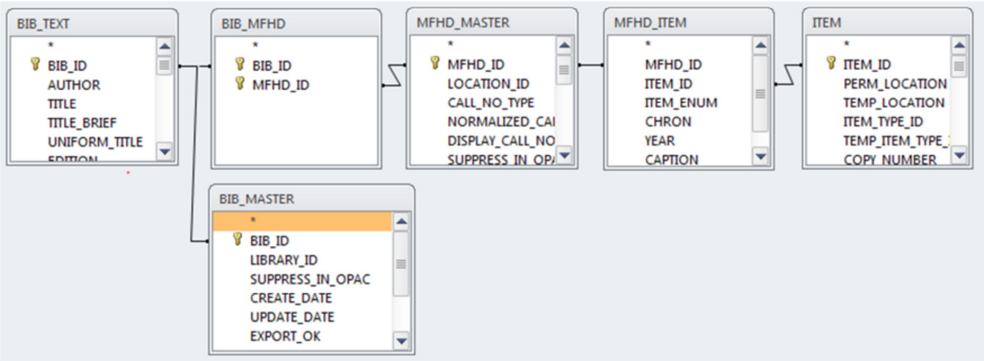
CARLI Consortium of Academic and Research Libraries in Illinois

I’m sure this is familiar to all of you.
The basic set of tables for getting from bibs to items.

And yet this backbone has some quirks.
Will it show your ebooks?
For most libraries, the answer is no
because you probably don’t create items for ebooks.

Let’s try adding some other tables to the backbone.

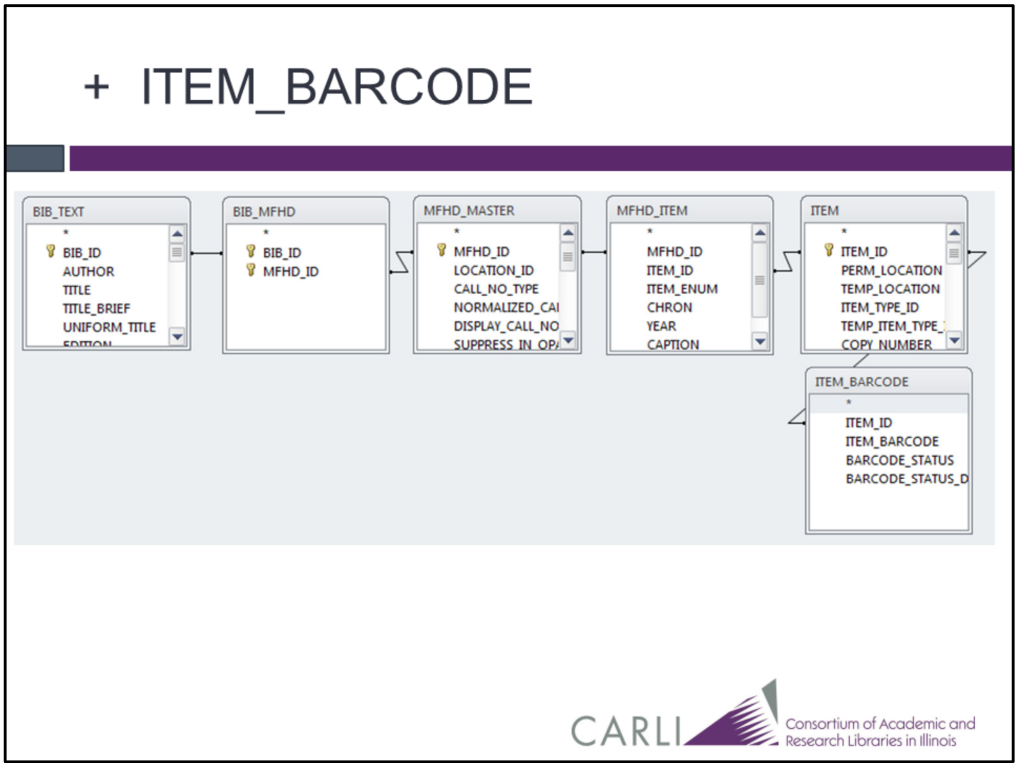
+ BIB_MASTER



CARLI Consortium of Academic and Research Libraries in Illinois

Suppose we’ve got all our criteria and columns set already
and we decide to add BIB_MASTER so we can include the bib CREATE_DATE.

Will the number of rows change?
No, because there’s always one BIB_TEXT record for every BIB_MASTER record.



+ ITEM_BARCODE

Suppose we add the ITEM_BARCODE table so we can include barcodes.
How might the number of rows change?

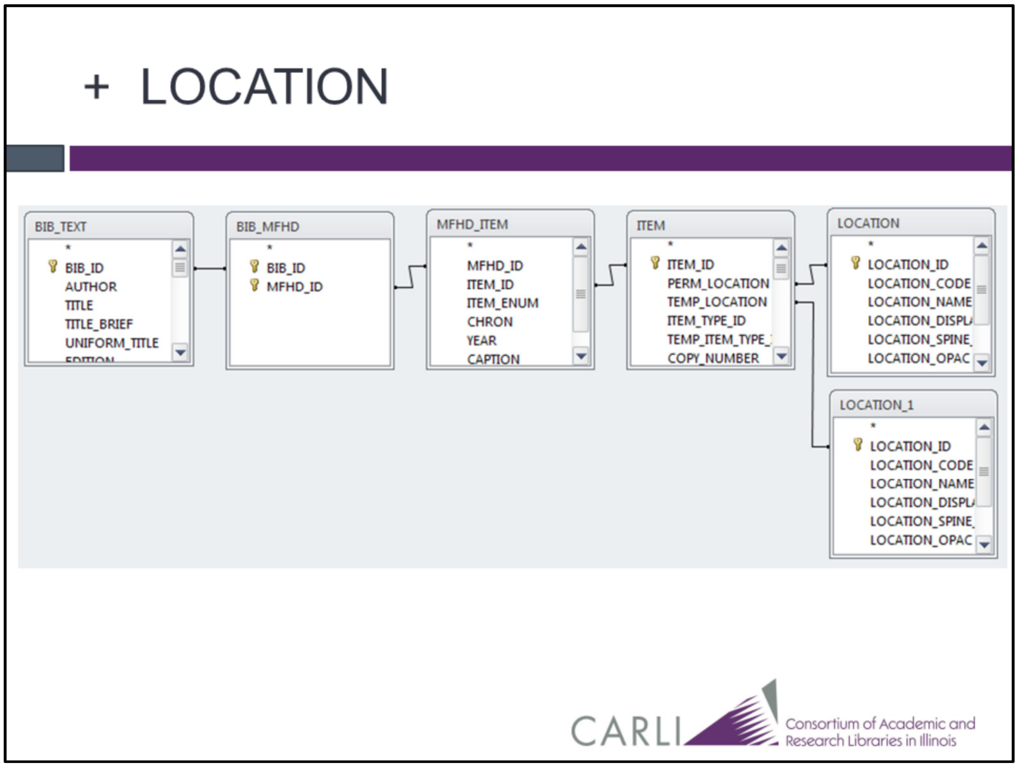
At your library, will microforms disappear?
Yes, if they're not barcoded.

And what would happen to items whose barcodes had to be replaced?
They'd be included twice.

These issues show up as duplicate or missing rows in list-of-stuff queries.
They show up as bad counts in counting queries.

So it's worthwhile to learn this stuff.
And it's even better to get control over it.

To get just the current barcodes, use BARCODE_STATUS = 1.
To keep Access from dropping unbarcoded items, you need an outer join from ITEM to ITEM_BARCODE.
We'll talk about the mechanics of creating an outer join a little later.
For now, I want to keep you focused on relationships between tables.



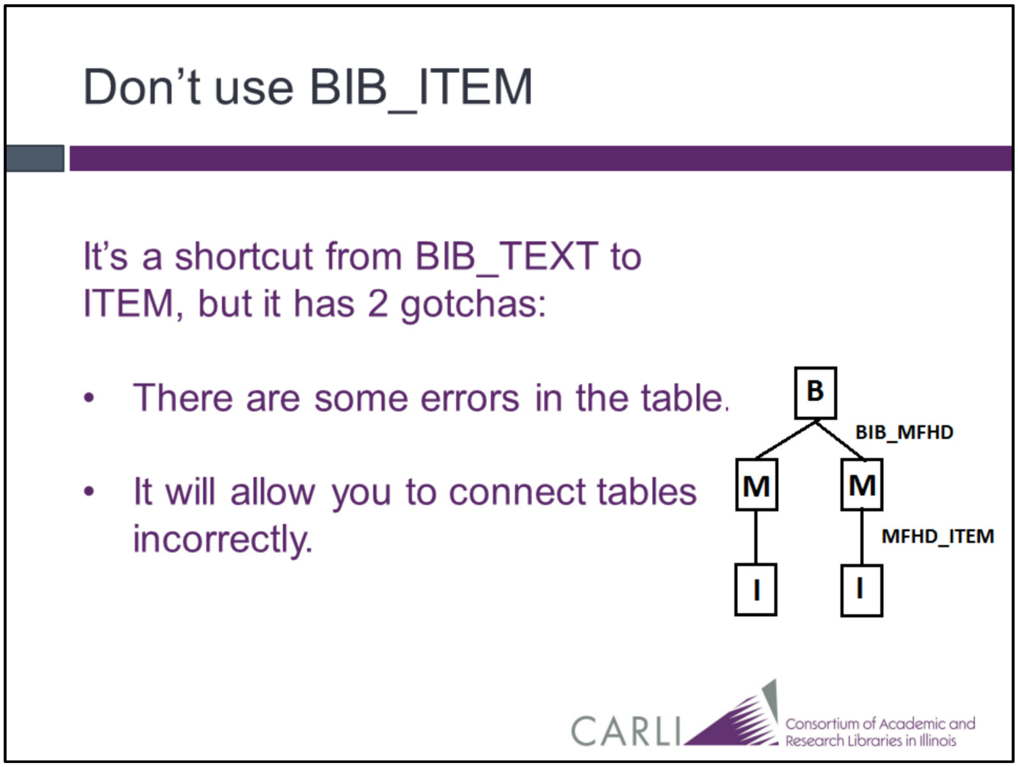
+ LOCATION

I've removed MFHD_MASTER in this example.
I don't absolutely need it unless I need one of the MFHD fields
Call#, MFHD Loc

And I've added the LOCATION table twice.
One linked to PERM_LOCATION and once to TEMP_LOCATION.
Notice that Access calls one of them LOCATION_1, so they can be distinguished.

I guess I'm trying to items with their perm and temp locations.
Which of my items will be included in the results?
Just the ones that have a temp location.

And if I want all of the items whether they have a temp location or not?
I need an outer join from ITEM to LOCATION_1.



Don’t use BIB_ITEM

It’s a shortcut from BIB_TEXT to ITEM, but it has 2 gotchas:

- There are some errors in the table.
- It will allow you to connect tables incorrectly.

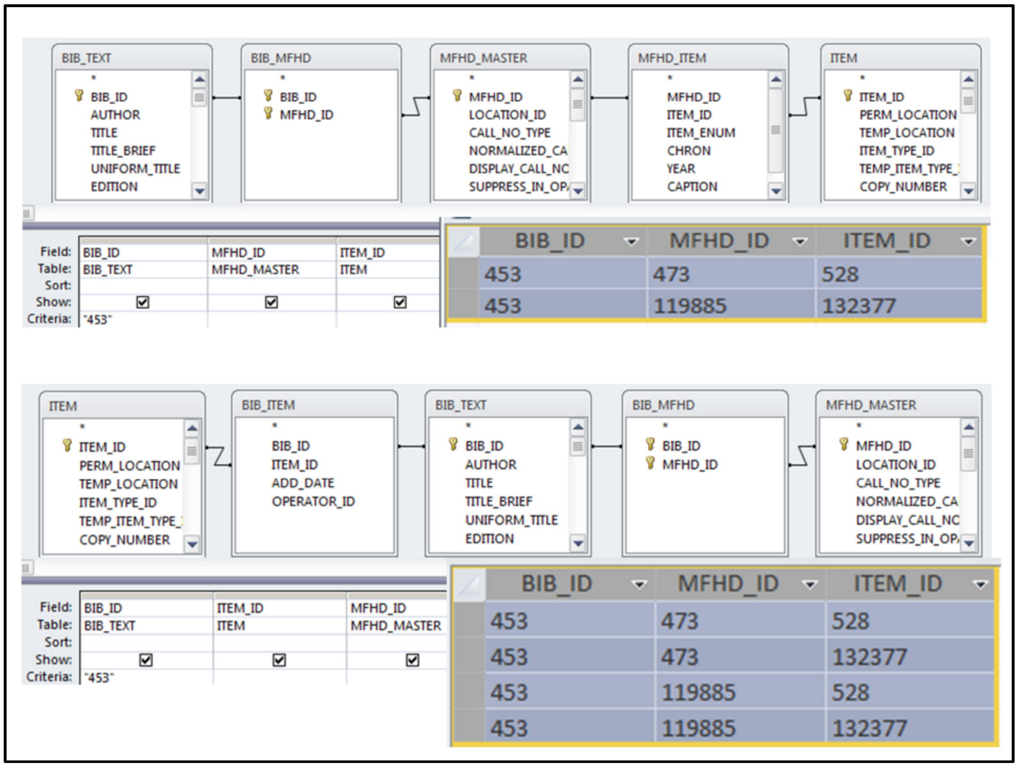
CARLI Consortium of Academic and Research Libraries in Illinois

The number of errors is not large
   but it’s enough to drive you crazy if you’re trying to get the numbers in several reports to jive.

And you won’t necessarily use it incorrectly.

But you don’t need it and I say, why worry?

Let’s consider this case, 2 copies of a title in 2 locations.
So we have one BIB_TEXT, 2 MFHD_MASTERs, and 2 ITEMS.
The lines represent the BIB_MFHD and MFHD_ITEM records.
How many rows of results are we expecting?



Field:	BIB_ID	MFHD_ID	ITEM_ID
Table:	BIB_TEXT	MFHD_MASTER	ITEM
Sort:			
Show:	☑	☑	☑
Criteria:	"453"		

BIB_ID	MFHD_ID	ITEM_ID
453	473	528
453	119885	132377

Field:	BIB_ID	ITEM_ID	MFHD_ID
Table:	BIB_TEXT	ITEM	MFHD_MASTER
Sort:			
Show:	☑	☑	☑
Criteria:	"453"		

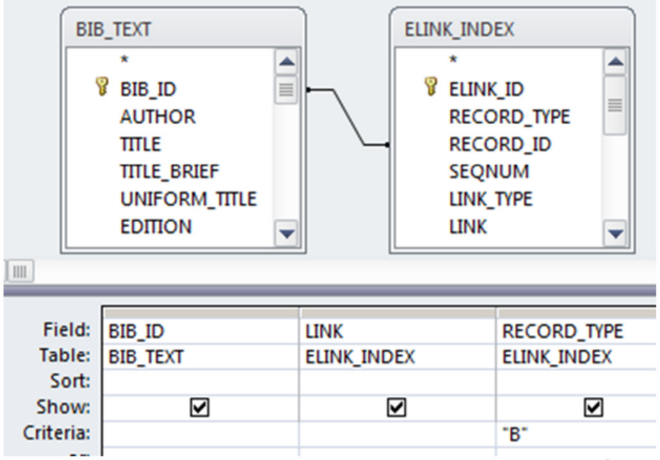
BIB_ID	MFHD_ID	ITEM_ID
453	473	528
453	473	132377
453	119885	528
453	119885	132377

I happen to know that bib 453 is an example of this.

Here are 2 queries.
The first uses the correct backbone.
The second uses BIB_ITEM.

We’re asking for BIB_ID, MDH_ID, and ITEM_ID,
so we’re expecting 2 rows of results, one for each item.

URLs in Bibs



Field:	BIB_ID	LINK	RECORD_TYPE
Table:	BIB_TEXT	ELINK_INDEX	ELINK_INDEX
Sort:			
Show:	☑	☑	☑
Criteria:			"B"

Here's another handy backbone.

The ELINK_INDEX table has a row for each URL in the database
from bibs, MFHDs, authorities, or eitems

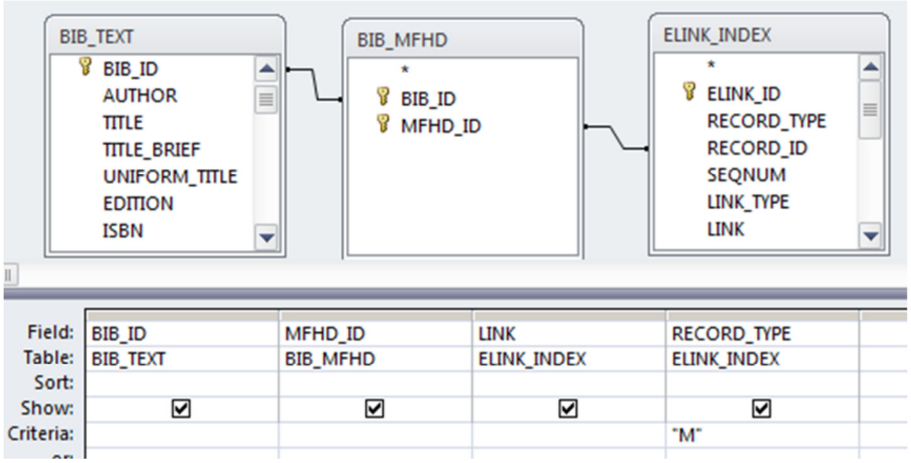
You may be surprised that I've joined BIB_ID to RECORD_ID.
Usually you join columns with the same name.

Since URLS, can be in bibs, MFHDs, authorities, or eitems
the RECORD_ID can be a BIB_ID, MFHD_ID, AUTH_ID, or EITEM_ID.

RECORD_TYPE is what keeps it straight.
B means the record_id is a bib_id.
A for AUTH_ID, M for MFHD_ID, E for EITEM_ID

So this little query lists all URLs on bibs.

URLs in MFHDs

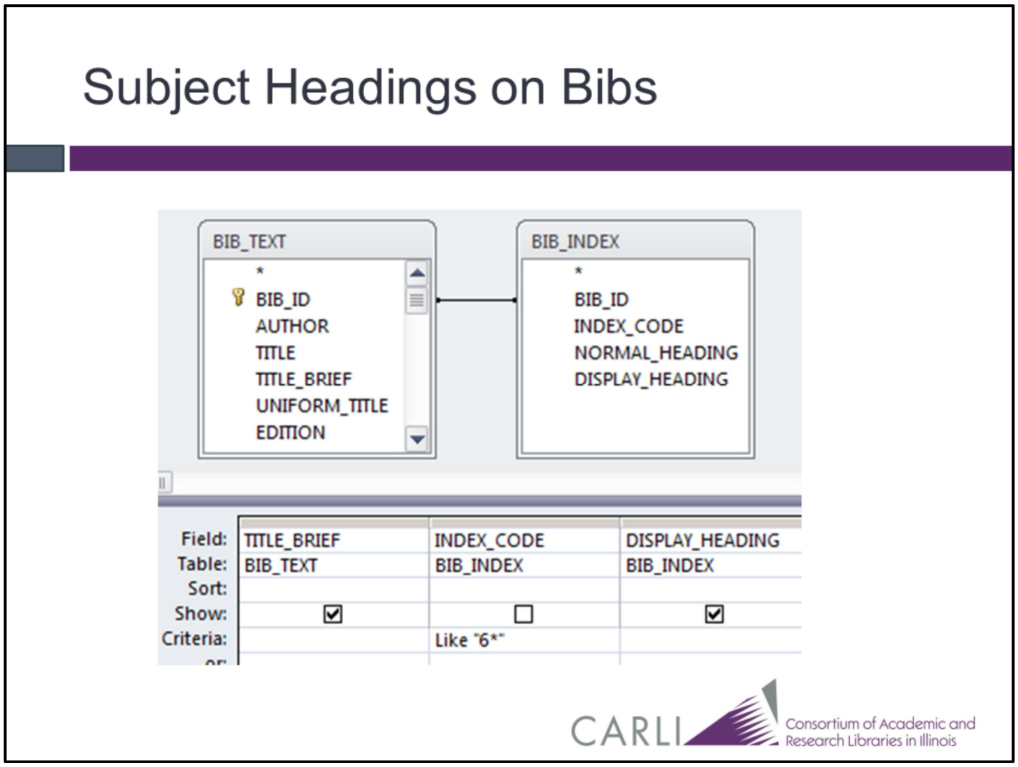


Field:	BIB_ID	MFHD_ID	LINK	RECORD_TYPE
Table:	BIB_TEXT	BIB_MFHD	ELINK_INDEX	ELINK_INDEX
Sort:				
Show:	☑	☑	☑	☑
Criteria:				"M"

CARLI Consortium of Academic and Research Libraries in Illinois

Similarly, this little query lists all the URLs in MFHDs.

Notice the link from MFHD_ID to RECORD_ID
And the "M" for MFHD in RECORD_TYPE.



Subject Headings on Bibs

BIB_TEXT
- *
- BIB_ID
- AUTHOR
- TITLE
- TITLE_BRIEF
- UNIFORM_TITLE
- EDITION

BIB_INDEX
- *
- BIB_ID
- INDEX_CODE
- NORMAL_HEADING
- DISPLAY_HEADING

Field:	TITLE_BRIEF	INDEX_CODE	DISPLAY_HEADING
Table:	BIB_TEXT	BIB_INDEX	BIB_INDEX
Sort:			
Show:	☑	□	☑
Criteria:		Like "6*"	
or:			

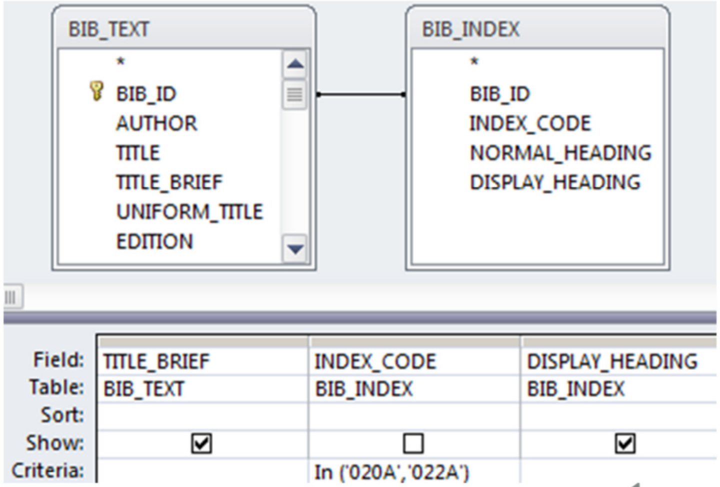
CARLI Consortium of Academic and Research Libraries in Illinois

Here’s another useful backbone.
It gets you all the subject headings on the bibs.

The BIB_INDEX table is where left-anchored indexes live.
INDEX_CODE tells you what type of index we’re searching.
You can see these in the SysAdmin client in the Searching section.
Most of the subject indexes begin with a 6, thus the criterion.

Notice that I un-checked the Show box so the index codes don’t clutter up my query.

All ISBNs or ISSNs

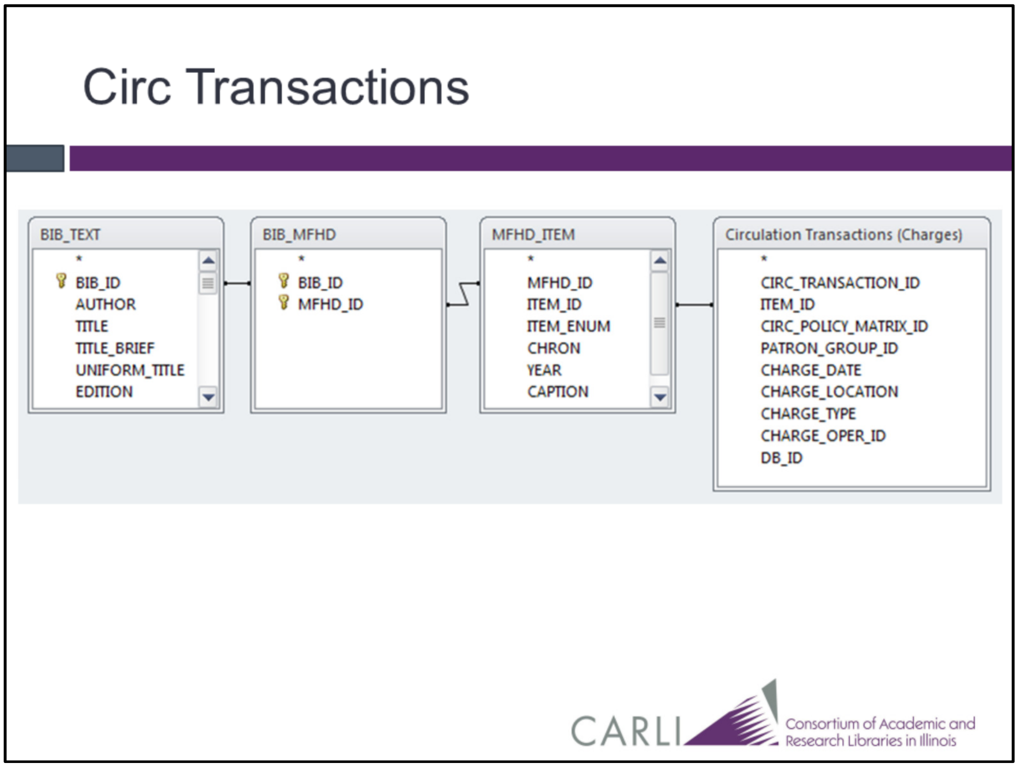


Field:	TITLE_BRIEF	INDEX_CODE	DISPLAY_HEADING
Table:	BIB_TEXT	BIB_INDEX	BIB_INDEX
Sort:			
Show:	☑	□	☑
Criteria:		In ('020A','022A')	

CARLI Consortium of Academic and Research Libraries in Illinois

Same tables, different criterion.

Now it’s giving me all the 020s and 022s.



Circ Transactions

BIB_TEXT
- *
- BIB_ID
- AUTHOR
- TITLE
- TITLE_BRIEF
- UNIFORM_TITLE
- EDITION

BIB_MFHD
- *
- BIB_ID
- MFHD_ID

MFHD_ITEM
- *
- MFHD_ID
- ITEM_ID
- ITEM_ENUM
- CHRON
- YEAR
- CAPTION

Circulation Transactions (Charges)
- *
- CIRC_TRANSACTION_ID
- ITEM_ID
- CIRC_POLICY_MATRIX_ID
- PATRON_GROUP_ID
- CHARGE_DATE
- CHARGE_LOCATION
- CHARGE_TYPE
- CHARGE_OPER_ID
- DB_ID

CARLI Consortium of Academic and Research Libraries in Illinois

Here’s a really useful circ backbone.

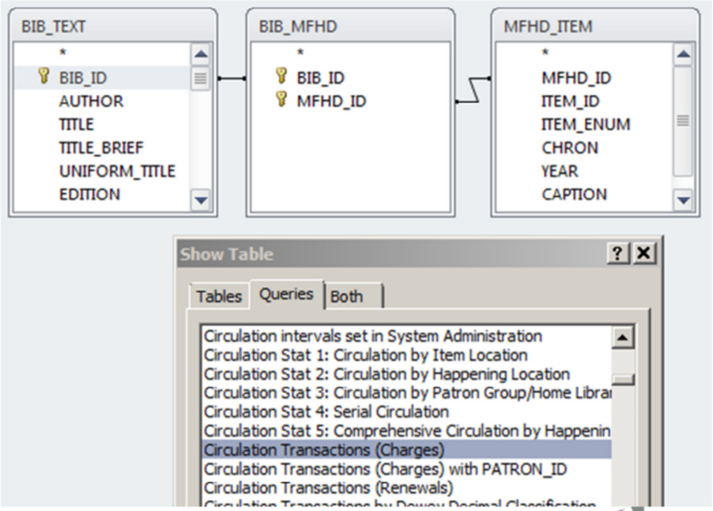
You probably know that CIRC_TRANSACTIONS is the current transactions
And CIRC_TRANS_ARCHIVE is the completed transactions.

Suppose you’re doing a weeding report and you want to know which items have circ’d.
You want to include the items that are in circ right now.
So you need data from both tables.
Circulation Transactions (Charges) does the job.

Will items that have never circ’d appear in this report? No.
And to get them?
An outer join.

Now Circulation Transactions (Charges) looks like a table
But it’s name isn’t in all caps like most of the Voyager tables.
It’s actually a subquery, but you don’t have to worry about that.

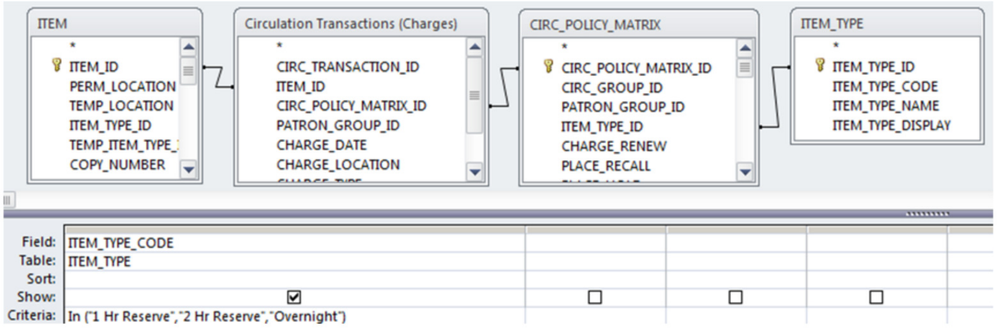
Finding Circulation Transactions (Charges)



Circulation intervals set in System Administration
Circulation Stat 1: Circulation by Item Location
Circulation Stat 2: Circulation by Happening Location
Circulation Stat 3: Circulation by Patron Group/Home Librai
Circulation Stat 4: Serial Circulation
Circulation Stat 5: Comprehensive Circulation by Happenin
Circulation Transactions (Charges)
Circulation Transactions (Charges) with PATRON_ID
Circulation Transactions (Renewals)
Circulation Transactions by Dewey Decimal Classification

Here I am, selecting the tables for my query.
I’ve got the Show Table window open.
It always opens on the Tables tab.
Click the Queries tab, scroll, and there it is.

Historic Reserves



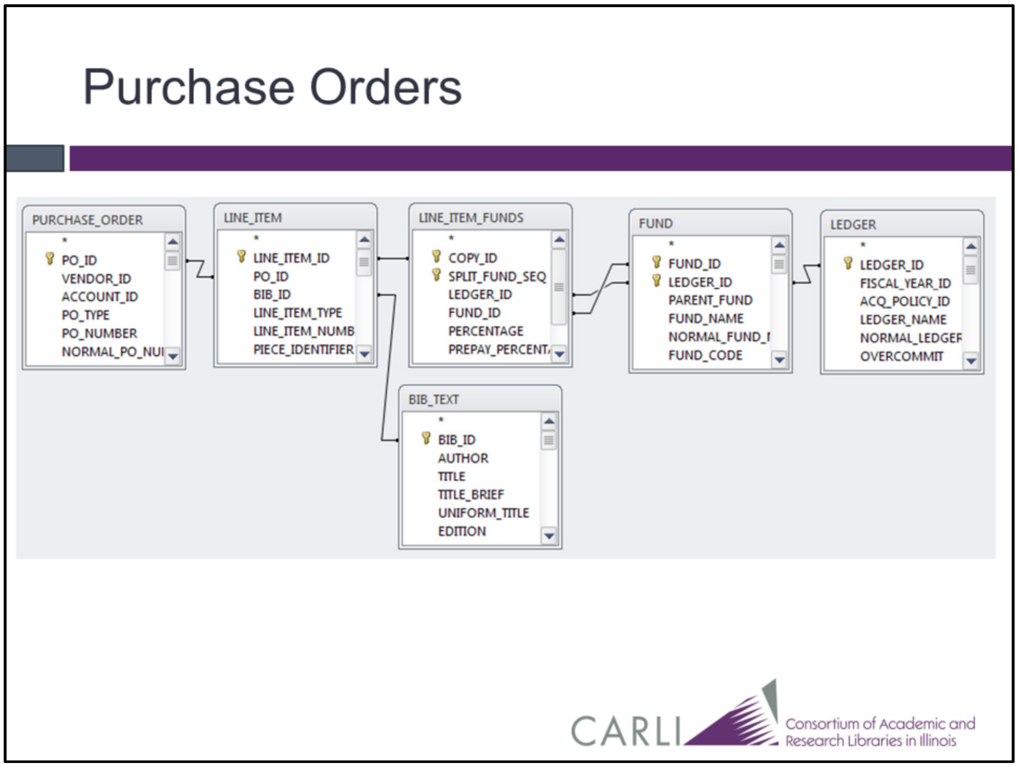
Field:	ITEM_TYPE_CODE			
Table:	ITEM_TYPE			
Sort:				
Show:	☑	□	□	□
Criteria:	In ("1 Hr Reserve","2 Hr Reserve","Overnight")			

CARLI Consortium of Academic and Research Libraries in Illinois

I find reserves queries confusing.
They get complicated when and item is on more than one reserve list
And some of the stats disappear when and item goes off reserve.

But reserves circ creates circ transaction records
And Voyager remembers the circ policy under when the charge occurred
And the policy includes the item type.

So this backbone give me circ
By the item type that the item had at the time the circ transaction was done.
And if you have distinctive item types for your reserve materials this is great.



Purchase Orders

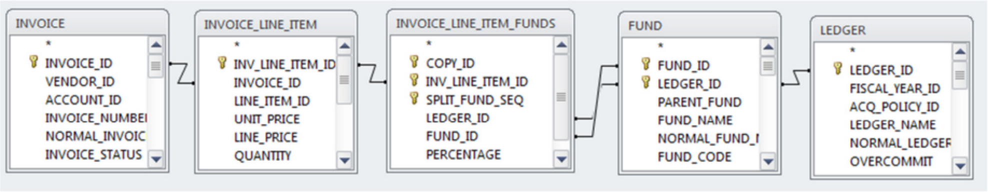
CARLI Consortium of Academic and Research Libraries in Illinois

Here’s the backbone for purchase orders.

Notice the double link.
Tables with funds always have to be double linked.
When you rollover funds, the FUND_ID doesn’t change.

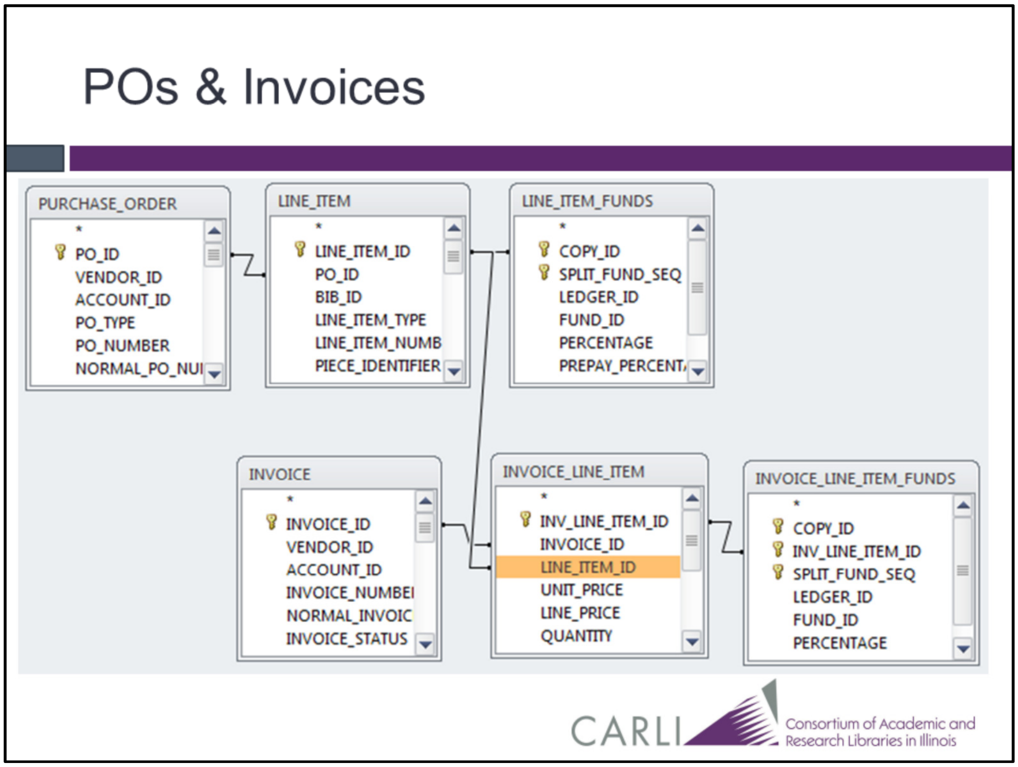
I stuck in the BIB_TEXT table.
It’s not really part of the backbone, but this is one place you get from POs and invoices to bib data.

Invoices



CARLI Consortium of Academic and Research Libraries in Illinois

The backbone for invoices is almost identical.



POs & Invoices

PURCHASE_ORDER
*
PO_ID
VENDOR_ID
ACCOUNT_ID
PO_TYPE
PO_NUMBER
NORMAL_PO_NUI

LINE_ITEM
*
LINE_ITEM_ID
PO_ID
BIB_ID
LINE_ITEM_TYPE
LINE_ITEM_NUMB
PIECE_IDENTIFIER

LINE_ITEM_FUNDS
*
COPY_ID
SPLIT_FUND_SEQ
LEDGER_ID
FUND_ID
PERCENTAGE
PREPAY_PERCENT/

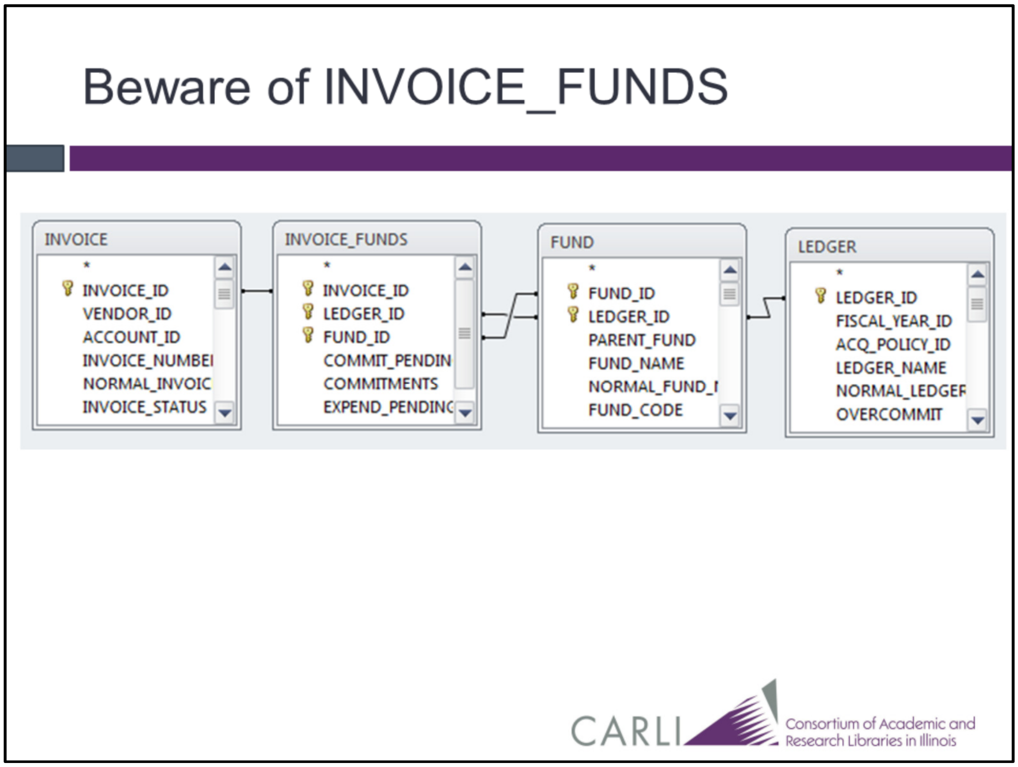
INVOICE
*
INVOICE_ID
VENDOR_ID
ACCOUNT_ID
INVOICE_NUMBEI
NORMAL_INVOIC
INVOICE_STATUS

INVOICE_LINE_ITEM
*
INV_LINE_ITEM_ID
INVOICE_ID
LINE_ITEM_ID
UNIT_PRICE
LINE_PRICE
QUANTITY

INVOICE_LINE_ITEM_FUNDS
*
COPY_ID
INV_LINE_ITEM_ID
SPLIT_FUND_SEQ
LEDGER_ID
FUND_ID
PERCENTAGE

CARLI Consortium of Academic and Research Libraries in Illinois

And to get between them, you need LINE_ITEM_ID in INVOICE_LINE_ITEM.



Beware of INVOICE_FUNDS

INVOICE
*
INVOICE_ID
VENDOR_ID
ACCOUNT_ID
INVOICE_NUMBEI
NORMAL_INVOIC
INVOICE_STATUS

INVOICE_FUNDS
*
INVOICE_ID
LEDGER_ID
FUND_ID
COMMIT_PENDIN
COMMITMENTS
EXPEND_PENDINC

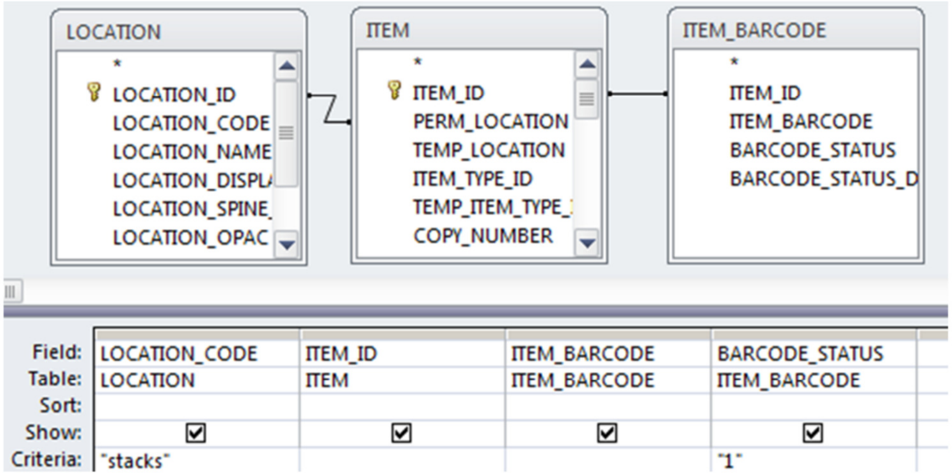
FUND
*
FUND_ID
LEDGER_ID
PARENT_FUND
FUND_NAME
NORMAL_FUND_I
FUND_CODE

LEDGER
*
LEDGER_ID
FISCAL_YEAR_ID
ACQ_POLICY_ID
LEDGER_NAME
NORMAL_LEDGEF
OVERCOMMIT

CARLI Consortium of Academic and Research Libraries in Illinois

This is a reasonable-looking backbone,
But INVOICE_FUNDS isn’t completely reliable.
Sometimes, when you delete an invoice, the totals go bad.
Then you have Ex Libris run the full fund repair job.

An Outer Join Example



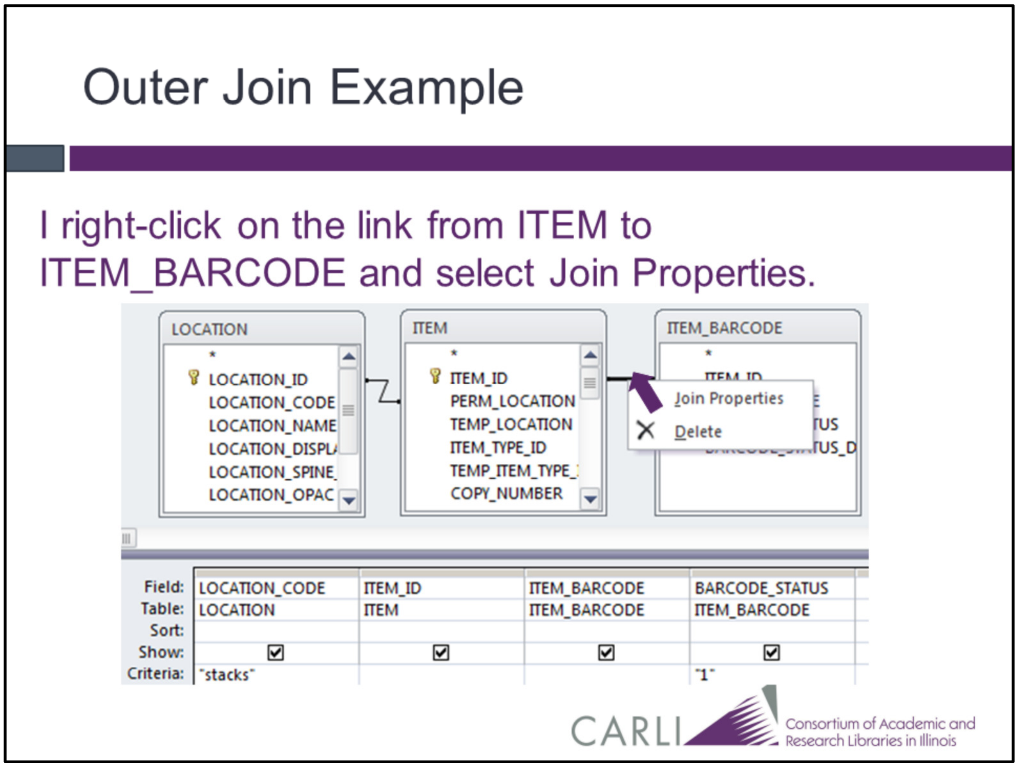
Field:	LOCATION_CODE	ITEM_ID	ITEM_BARCODE	BARCODE_STATUS
Table:	LOCATION	ITEM	ITEM_BARCODE	ITEM_BARCODE
Sort:				
Show:	☑	☑	☑	☑
Criteria:	"stacks"			"1"

CARLI Consortium of Academic and Research Libraries in Illinois

I promised you some outer joins a while ago.

Let’s say I want a list of items in a location, and I want their item_ids and barcodes.

What happens to the unbarcoded items?
They get dropped.
To fix this, I need an outer join from ITEM to ITEM_BARCODE.



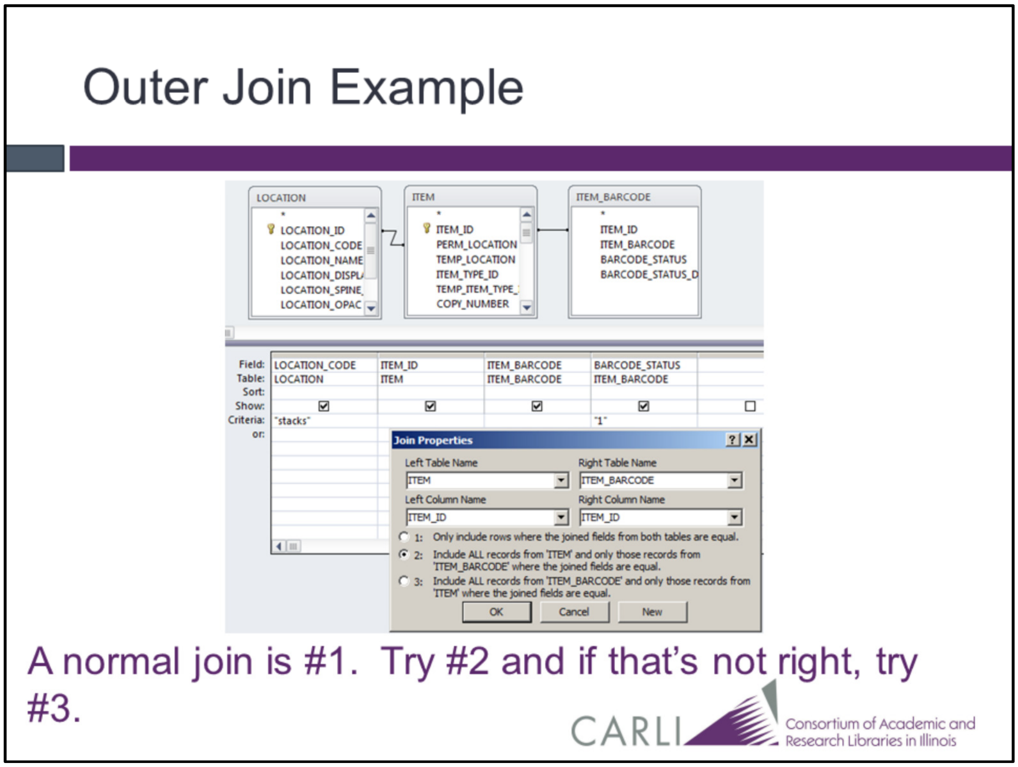
Outer Join Example

I right-click on the link from ITEM to ITEM_BARCODE and select Join Properties.

Field:	LOCATION_CODE	ITEM_ID	ITEM_BARCODE	BARCODE_STATUS
Table:	LOCATION	ITEM	ITEM_BARCODE	ITEM_BARCODE
Sort:				
Show:	☑	☑	☑	☑
Criteria:	"stacks"			"1"

CARLI Consortium of Academic and Research Libraries in Illinois

If you get a different dialog box, you clicked on the wrong place. Try again.



Outer Join Example

Field:	LOCATION_CODE	ITEM_ID	ITEM_BARCODE	BARCODE_STATUS	
Table:	LOCATION	ITEM	ITEM_BARCODE	ITEM_BARCODE	
Sort:					
Show:	☑	☑	☑	☑	☐
Criteria:	"stacks"			"1"	
or:					

Join Properties

Left Table Name: ITEM

Right Table Name: ITEM_BARCODE

Left Column Name: ITEM_ID

Right Column Name: ITEM_ID

○ 1: Only include rows where the joined fields from both tables are equal.

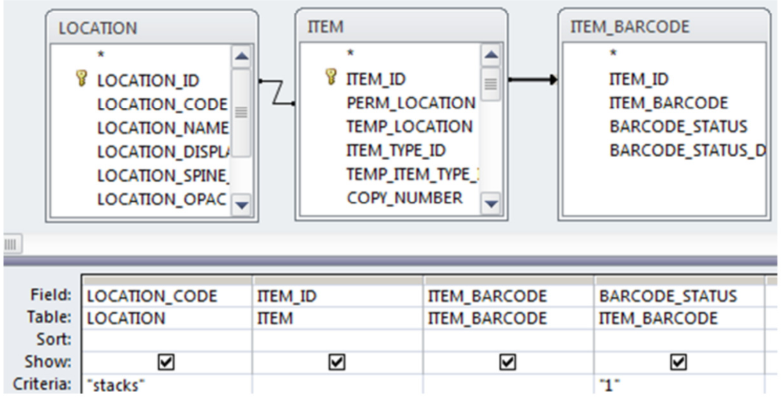
● 2: Include ALL records from 'ITEM' and only those records from 'ITEM_BARCODE' where the joined fields are equal.

○ 3: Include ALL records from 'ITEM_BARCODE' and only those records from 'ITEM' where the joined fields are equal.

A normal join is #1. Try #2 and if that's not right, try #3.

You want to get an arrow from ITEM to ITEM_BARCODE

Outer Join Example



Field:	LOCATION_CODE	ITEM_ID	ITEM_BARCODE	BARCODE_STATUS
Table:	LOCATION	ITEM	ITEM_BARCODE	ITEM_BARCODE
Sort:				
Show:	☑	☑	☑	☑
Criteria:	"stacks"			"1"

The arrow says, "When we go from ITEM to ITEM_BARCODE, if there are no matching ITEM_IDs, don't drop the row."

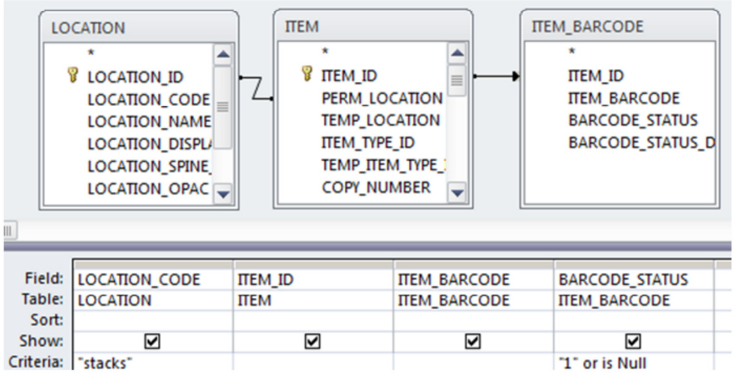
CARLI Consortium of Academic and Research Libraries in Illinois

The direction matters.

Now what value will be in the ITEM_BARCODE and BARCODE_STATUS columns for an unbarcoded item?
Null, a special value that means no value has been set.
It's different from blank or zero.

So what's the criterion in BARCODE_STATUS going to do?
It will drop the unbarcoded items because "1" <> Null.

An Outer Join Example



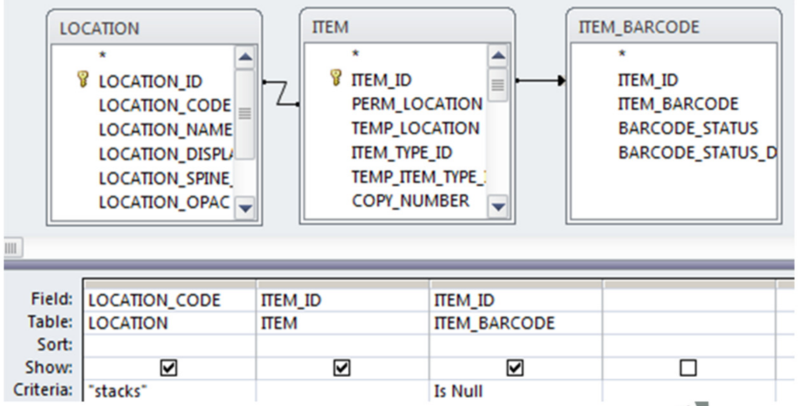
Field:	LOCATION_CODE	ITEM_ID	ITEM_BARCODE	BARCODE_STATUS
Table:	LOCATION	ITEM	ITEM_BARCODE	ITEM_BARCODE
Sort:				
Show:	☑	☑	☑	☑
Criteria:	"stacks"			"1" or is Null

I need to correct the criterion on BARCODE_STATUS so the unbarcoded items will be included.

CARLI Consortium of Academic and Research Libraries in Illinois

An Outer Join Example

Now suppose I want to find the unbarcoded items in the stacks. I can use Is Null again.



Field:	LOCATION_CODE	ITEM_ID	ITEM_ID	
Table:	LOCATION	ITEM	ITEM_BARCODE	
Sort:				
Show:	☑	☑	☑	☐
Criteria:	"stacks"		Is Null	

CARLI Consortium of Academic and Research Libraries in Illinois

To find unbarcoded items…
I deleted the ITEM_BARCODE and BARCODE_STATUS columns from ITEM_BARCODE, chose ITEM_ID instead,
And put the Is Null criterion on that column.

Voila!

Why did I choose to put Is Null on ITEM_ID?
All fields in a missing record pointed to by an outer join are Null,
But there are cases where other fields are Null.
I don't want to worry about this.
So I always put the Is Null on the column the arrow points to.