



ALL THE ANSWERS WITH ADVANCED POLARIS SQL

PRESENTER: RICHARD KENIG

WELCOME TO IUG 2016

INTRO AND OUTLINE

- Polaris Database Structure
- Most Commonly Used Tables
- Transaction Database
- Using Your Own Database
- Building Reports

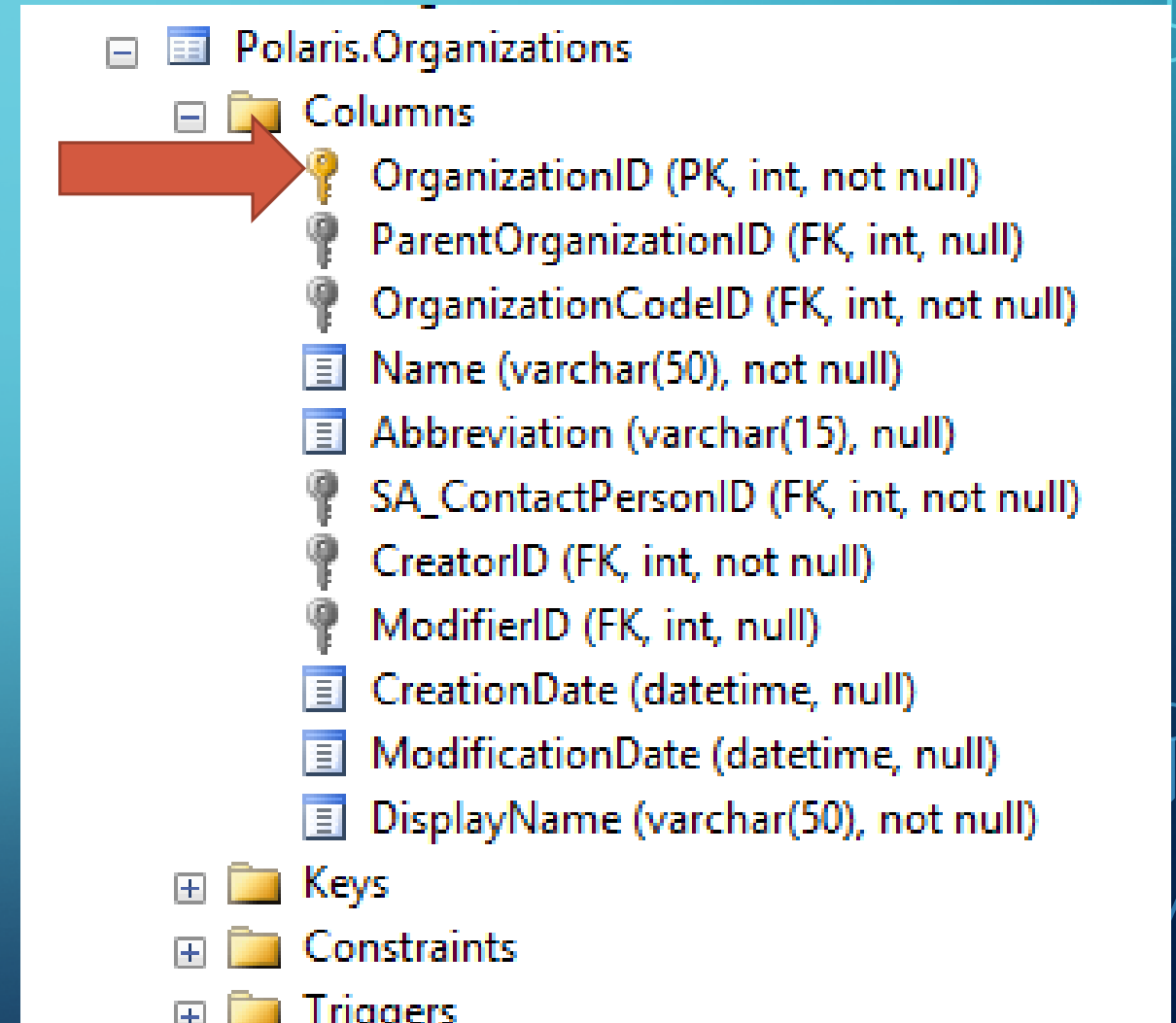
POLARIS DATABASE STRUCTURE

- TableID Field
- Dependencies
- Stored Procedures
- Functions
- Jobs

DATABASE STRUCTURE

TABLE ID FIELD

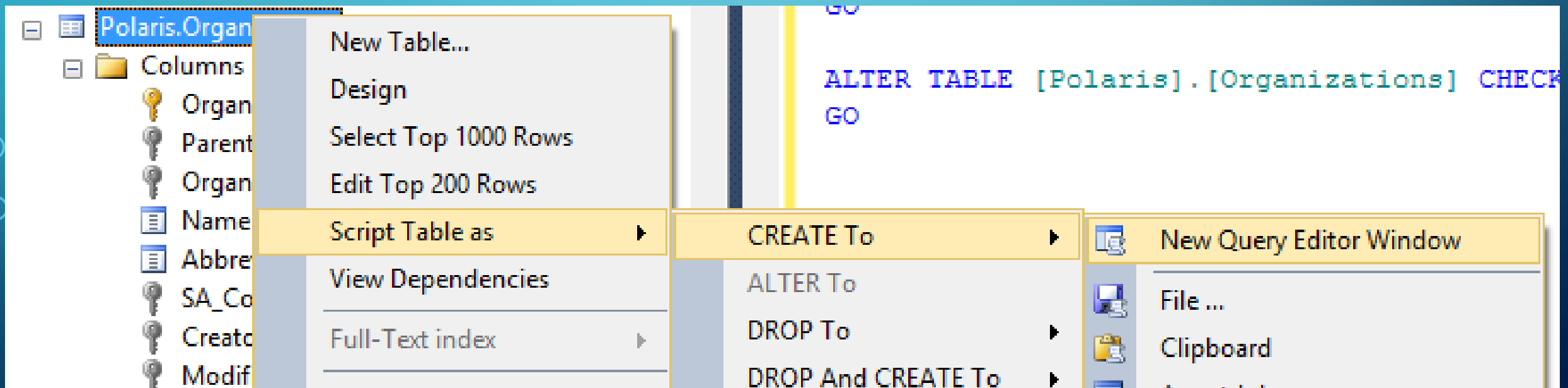
- Each table has a unique ID field
- Always (with few exceptions) is formatted <tablename>ID



DATABASE STRUCTURE

TABLEID FIELD

- Not 100% - to view details



DATABASE STRUCTURE

TABLEID FIELD

- Code View

```
ALTER TABLE [Polaris].[Organizations] WITH CHECK  
ADD CONSTRAINT [fk_OrgHierarchy]  
FOREIGN KEY([ParentOrganizationID])  
REFERENCES [Polaris].[Organizations] ([OrganizationID])  
GO
```

DATABASE STRUCTURE

TABLEID FIELD

- Code View

```
ALTER TABLE [Polaris].[Organizations] WITH CHECK  
ADD CONSTRAINT [fk_OrgHierarchy] ← Name of Key  
FOREIGN KEY([ParentOrganizationID])  
REFERENCES [Polaris].[Organizations] ([OrganizationID])  
GO
```

DATABASE STRUCTURE

TABLEID FIELD

- Code View

```
ALTER TABLE [Polaris].[Organizations] WITH CHECK
ADD CONSTRAINT [fk_OrgHierarchy] ← Name of Key
FOREIGN KEY([ParentOrganizationID]) ← Name of Column in table
REFERENCES [Polaris].[Organizations] ([OrganizationID])
GO
```


DATABASE STRUCTURE

TABLEID FIELD

- Code View

```
ALTER TABLE [Polaris].[Organizations] WITH CHECK
ADD CONSTRAINT [fk_OrgHierarchy]
FOREIGN KEY([ParentOrganizationID])
REFERENCES [Polaris].[Organizations] ([OrganizationID])
GO
```

Annotations:

- Name of Key**: [fk_OrgHierarchy]
- Name of Column in table**: [ParentOrganizationID]
- Name of Referenced Table**: [Polaris].[Organizations]
- Referenced Column**: [OrganizationID]

DATABASE STRUCTURE

DEPENDENCIES

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the 'Database Structure' pane shows a tree view of the database objects. The 'Polaris.Organization' table is selected, and a context menu is open with the 'View Dependencies' option highlighted. A red arrow points from this menu item to the 'Object Dependencies - Organizations' window on the right.

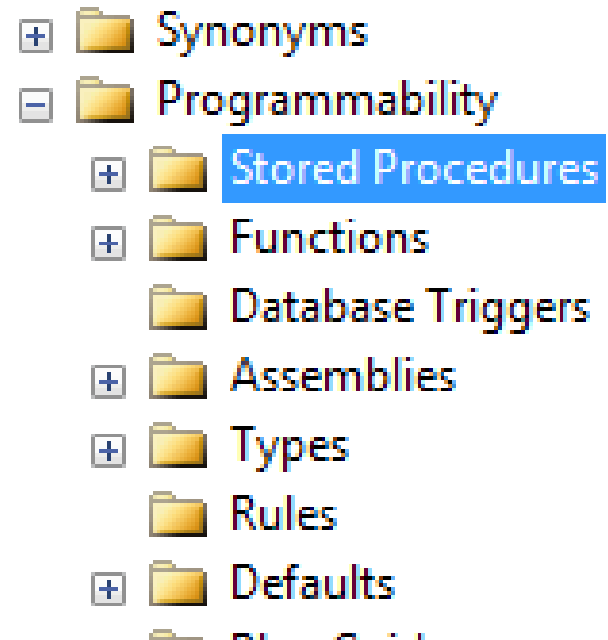
The 'Object Dependencies - Organizations' window shows the following information:

- Select a page:** General
- Script** (icon) **Help** (icon)
- Objects that depend on [Organizations]** (selected) **Objects on which [Organizations] depends** (unselected)
- Dependencies:**
 - Organizations
 - Acq_CheckDupePOLISegments
 - Acq_CheckItemTemplReqFields
 - Acq_CheckPORelReqFields
 - Acq_CreateNewInvoice
 - Acq_GetDupInvSegments
 - Acq_GetDuplicateINLISegments
 - Acq_GetDuplicatePOLISegments
 - Acq_GetEDI850EnhancedLineData
 - Acq_GetExchangeRateDetails
 - Acq_GetInvoicesLinkedToInv
 - Acq_GetLinkedCopyRecords
 - Acq_GetLinkedFunds
 - Acq_GetLinkedPOLineItemSegmentData
 - Acq_GetLinkedSelectionLists
 - Acq_GetLinkedSOParts
 - Acq_GetNumberOfOrgsAndItemTemplates
 - Acq_GetOrderTemplateProperties
 - Acq_GetOrgAndParentFunds
 - Acq_GetOrgAndParentOpenFunds
- Connection:**
 - Server: trac-prod
 - Connection: MLSCAL/richardk
 - [View connection properties](#)
- Progress**
- Selected object Name:** [TRAC-PROD].[Polaris].[Polaris].[Organizations]

DATABASE STRUCTURE

STORED PROCEDURES

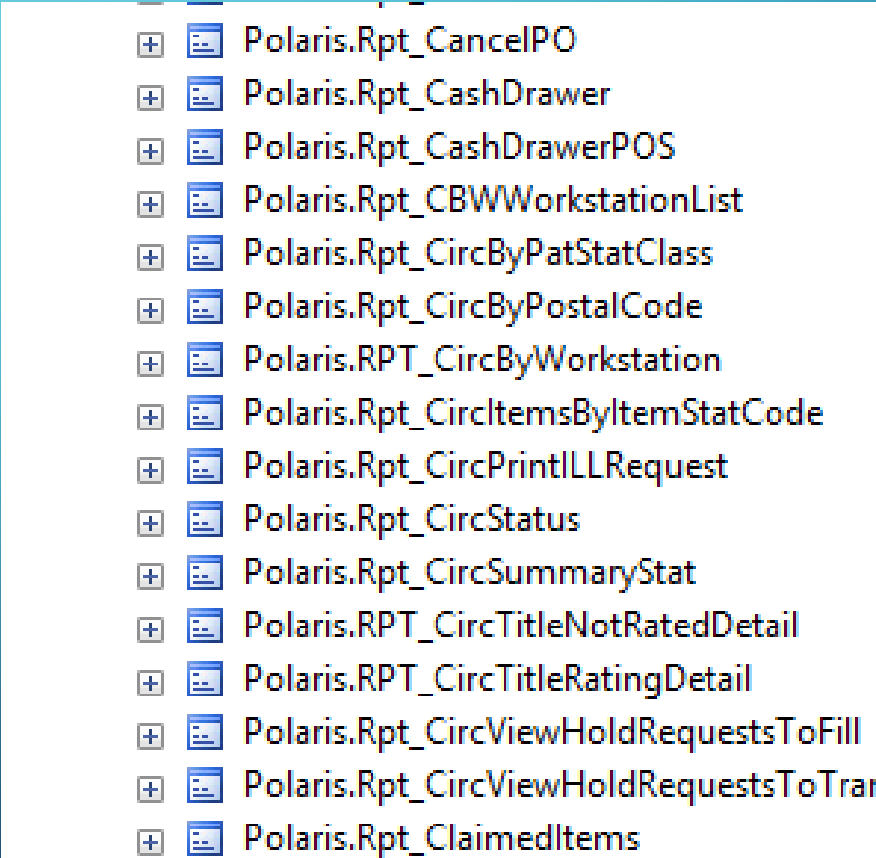
- Used to perform complex tasks
- Can return results
- Can be scheduled
- Can be used to insert or modify data in the database
- Can be given variables during runtime



DATABASE STRUCTURE

STORED PROCEDURES

- Used to perform complex tasks
- Can return results
- Can be scheduled
- Can be used to insert or modify data in the database
- Can be given variables during runtime



A screenshot of a database management tool's interface, specifically a list of stored procedures. The list is displayed in a table-like format with a light blue header. Each row contains a small icon (a plus sign in a square) followed by the procedure name. The procedure names are all prefixed with 'Polaris.Rpt_'. The list includes:

- Polaris.Rpt_CancelPO
- Polaris.Rpt_CashDrawer
- Polaris.Rpt_CashDrawerPOS
- Polaris.Rpt_CBWorkstationList
- Polaris.Rpt_CircByPatStatClass
- Polaris.Rpt_CircByPostalCode
- Polaris.RPT_CircByWorkstation
- Polaris.Rpt_CircItemsByItemStatCode
- Polaris.Rpt_CircPrintILLRequest
- Polaris.Rpt_CircStatus
- Polaris.Rpt_CircSummaryStat
- Polaris.RPT_CircTitleNotRatedDetail
- Polaris.RPT_CircTitleRatingDetail
- Polaris.Rpt_CircViewHoldRequestsToFill
- Polaris.Rpt_CircViewHoldRequestsToTrans
- Polaris.Rpt_ClaimedItems

STORED PROCEDURES

-
- The screenshot shows a list of stored procedures in SQL Server Enterprise Manager. The procedure 'Polaris.Rpt_StatisticalSummaryCirc...' is selected, and a context menu is open. The menu options are: 'New Stored Procedure...', 'Modify', 'Execute Stored Procedure...', 'Script Stored Procedure as...', 'View Dependencies', 'Policies', 'Facets', 'Start PowerShell', and 'Reports'. A green arrow points to the 'Modify' option.

DATABASE STRUCTURE

STORED PROCEDURES

- Used to perform complex tasks
- Can return results
- Can be scheduled
- Can be used to insert or modify data in the database
- Can be given variables during runtime



DATABASE STRUCTURE

STORED PROCEDURES

- Can be given variables during runtime

```
ALTER PROCEDURE [Polaris].[Rpt_StatisticalSummaryCircs]
    @dtBeginDate as datetime, -- Beginning date range
    @dtEndDate as datetime,    -- Ending date range
    @OrganizationList as varchar(max) -- OrganizationIDs that was selected.
AS
SET NOCOUNT ON
```

DATABASE STRUCTURE

STORED PROCEDURES

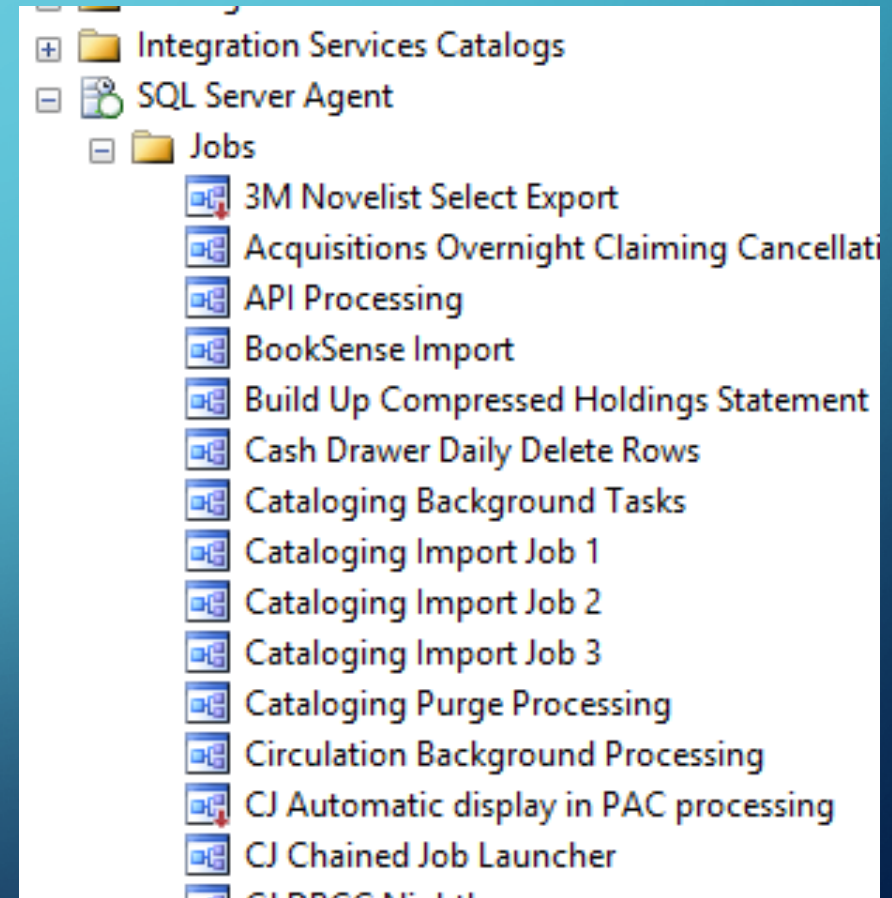
- Can reference or run other procedures

```
SET @SQLStmt = 'INSERT INTO #TempCircs EXECUTE Polaris.Rpt_StatisticalSummaryCircs ''' + @cBeginDate + ''';  
exec sp_executesql @SQLStmt  
  
SET @SQLStmt = 'INSERT INTO #TempFinancials EXECUTE Polaris.Rpt_StatisticalSummaryFinancials ''' + @cBeginDate + ''';  
exec sp_executesql @SQLStmt
```

DATABASE STRUCTURE

JOBS

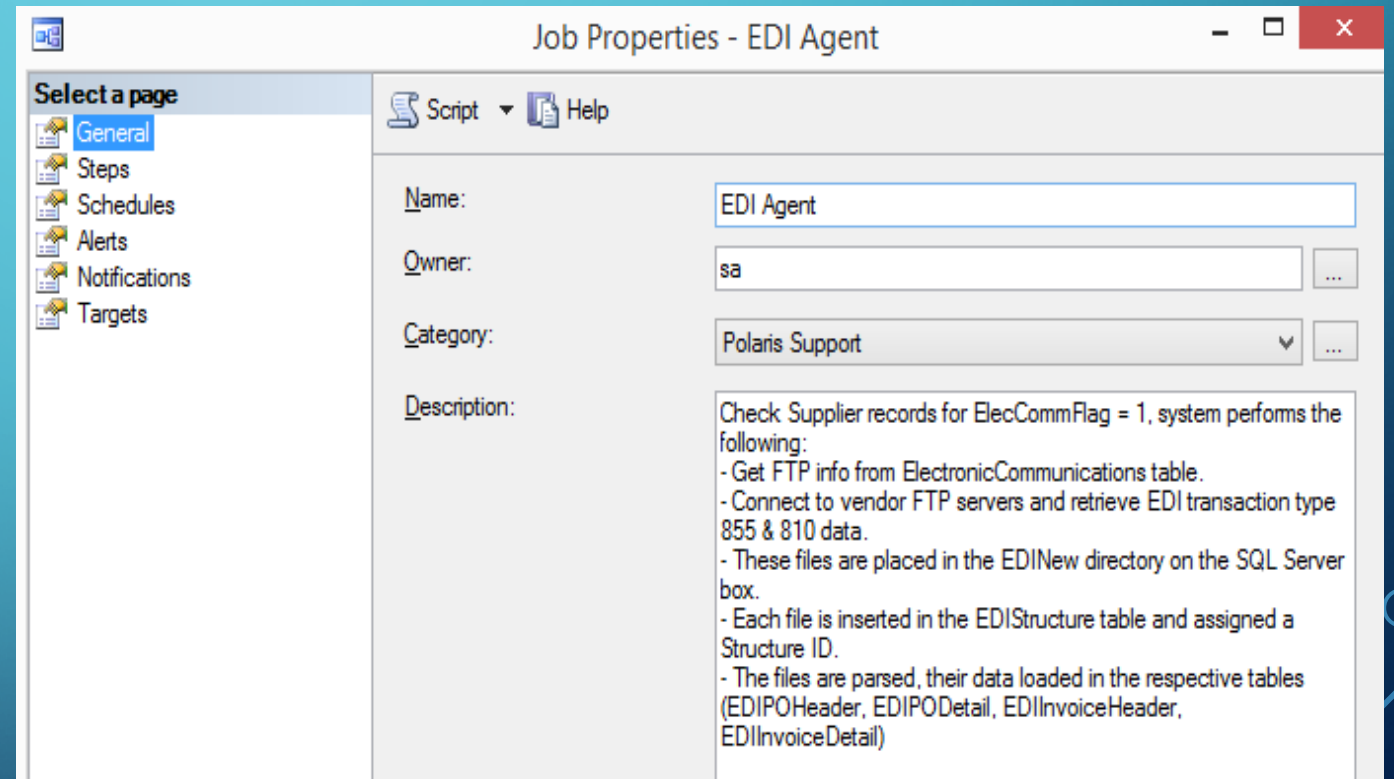
- Perform scheduled activities
- Can be coded directly, or call procedures and functions
- Used extensively for any regular task



DATABASE STRUCTURE

JOB CREATION

- Multiple parts to a job
- Properties and description
- Steps to perform
- Schedules
- Alerts
- Notifications



The screenshot shows a Windows-style window titled "Job Properties - EDI Agent". On the left is a "Select a page" sidebar with icons and labels for "General", "Steps", "Schedules", "Alerts", "Notifications", and "Targets". The "General" page is selected. The main area contains fields for "Name", "Owner", "Category", and "Description".

Field	Value
Name	EDI Agent
Owner	sa
Category	Polaris Support
Description	Check Supplier records for ElecCommFlag = 1, system performs the following: - Get FTP info from ElectronicCommunications table. - Connect to vendor FTP servers and retrieve EDI transaction type 855 & 810 data. - These files are placed in the EDINew directory on the SQL Server box. - Each file is inserted in the EDIStructure table and assigned a Structure ID. - The files are parsed, their data loaded in the respective tables (EDIPOHeader, EDIPODetail, EDIInvoiceHeader, EDIInvoiceDetail)

DATABASE STRUCTURE

JOB CREATION

- Multiple parts to a job
- Properties and description
- Steps to perform
- Schedules
- Alerts
- Notifications

The screenshot displays two windows from SQL Server Enterprise Manager. The top window, 'Job Properties - EDI Agent', shows the 'Steps' page selected in the left-hand 'Select a page' pane. The 'Job step list' table contains one entry:

St...	Name	Type	On Success	On Failure
1	step1	Transact...	Quit the j...	Quit the job...

The bottom window, 'Job Step Properties - step1', shows the 'Advanced' page selected. It contains the following configuration details:

- Step name:** step1
- Type:** Transact-SQL script (T-SQL)
- Run as:** (empty dropdown)
- Database:** Polaris
- Command:** EXEC Polaris.Polaris.EDI_ExecuteAgent

Red arrows indicate the flow from the 'Steps' page in the top window to the 'Job Step Properties' window, and from the 'Advanced' page in the bottom window to the 'Command' field.

DATABASE STRUCTURE

JOB CREATION

- Multiple parts to a job
- Properties and description
- Steps to perform
- Schedules
- Alerts
- Notifications

The screenshot shows the 'Job Properties - EDI Agent' window. The 'Schedules' tab is selected in the left-hand 'Select a page' menu. The 'Schedule list' table contains one entry:

ID	Name	Enabled	Description
29	Polaris Default	Yes	Occurs every day at 8:00:00 AM. Sch

The 'Job Schedule Properties - Polaris Default' dialog is open, showing the configuration for the selected schedule. The 'Name' field is 'Polaris Default'. The 'Schedule type' is 'Recurring'. The 'Enabled' checkbox is checked. The 'One-time occurrence' section is collapsed. The 'Frequency' section shows 'Occurs: Daily', 'Recurs every: 1 day(s)'. The 'Daily frequency' section shows 'Occurs once at: 8:00:00 AM'. The 'Duration' section shows 'Start date: 09/08/2001' and 'End date: 24/02/2016'. The 'Summary' section is at the bottom.

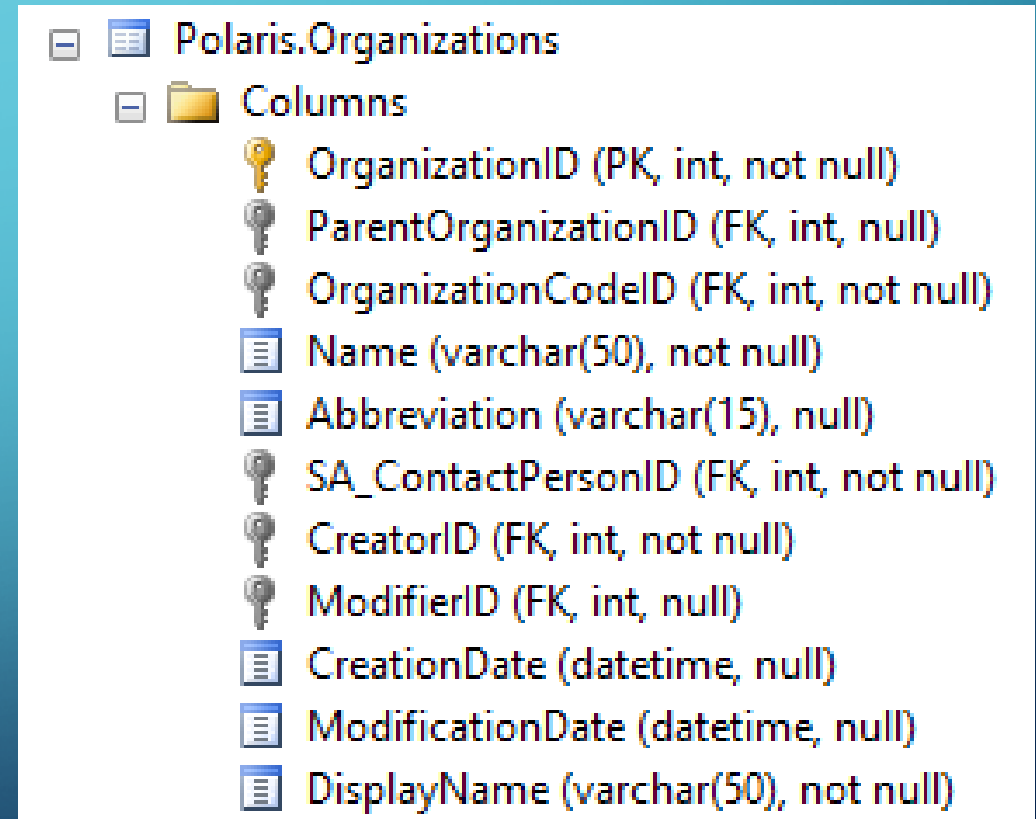
MOST COMMONLY USED TABLES

- Organizations
- Items
- Patrons
- Accounting
- Bibliographic

MOST COMMONLY USED TABLES

ORGANIZATIONS

- Most frequently used table
- In most reports
 - As a variable
 - As a way of splitting data
- Polaris.Polaris.Organizations



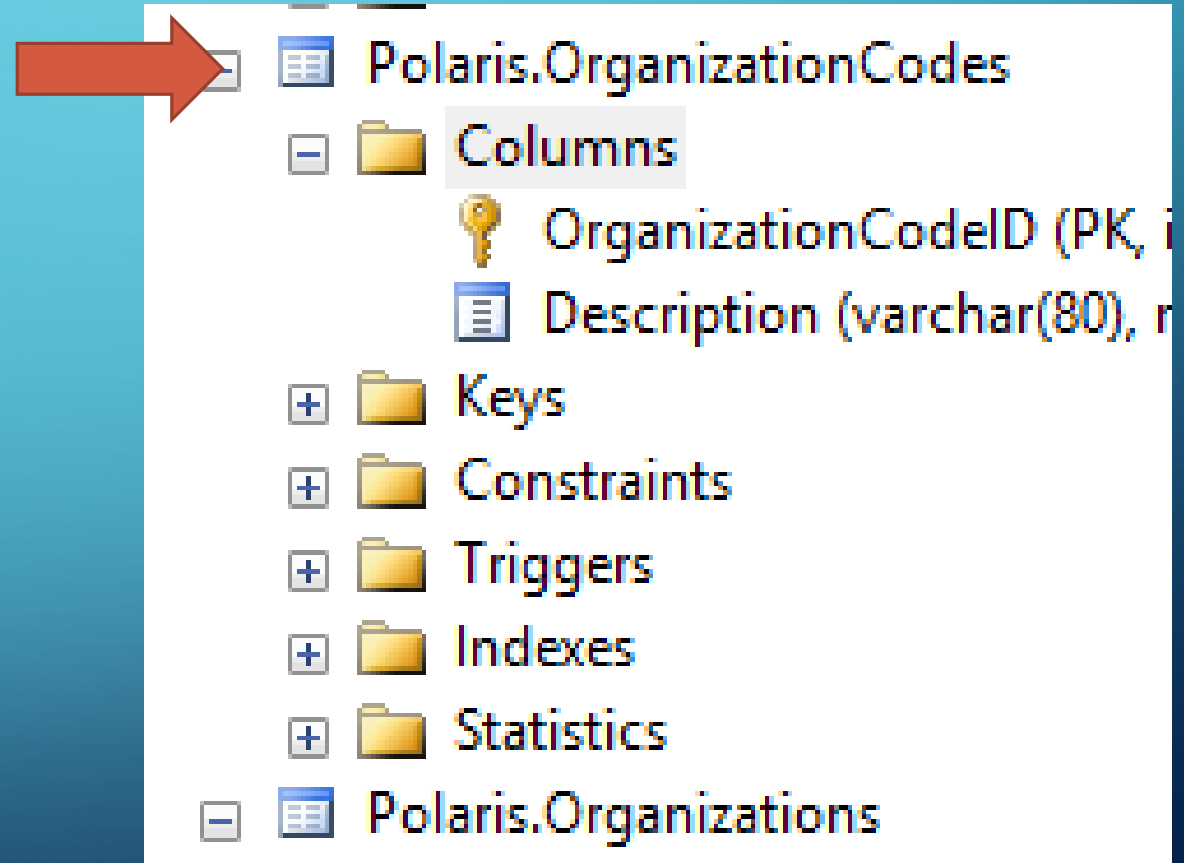
The screenshot displays a database schema viewer for the 'Polaris.Organizations' table. It shows a list of columns with their respective data types, constraints, and key indicators (primary key, foreign key).

Polaris.Organizations	
Columns	
OrganizationID	(PK, int, not null)
ParentOrganizationID	(FK, int, null)
OrganizationCodeID	(FK, int, not null)
Name	(varchar(50), not null)
Abbreviation	(varchar(15), null)
SA_ContactPersonID	(FK, int, not null)
CreatorID	(FK, int, not null)
ModifierID	(FK, int, null)
CreationDate	(datetime, null)
ModificationDate	(datetime, null)
DisplayName	(varchar(50), not null)

MOST COMMONLY USED TABLES

ORGANIZATIONS

- Most frequently used table
- In most reports
 - As a variable
 - As a way of splitting data
- Polaris.Polaris.Organizations



MOST COMMONLY USED TABLES

ORGANIZATIONS

- Code Snippet for joining tables

```
= SELECT *  
FROM Polaris.Organizations PO  
JOIN Polaris.OrganizationCodes POC ON  
POC.OrganizationCodeID = PO.OrganizationCodeID
```

MOST COMMONLY USED TABLES

ORGANIZATIONS

Polaris.Organizations	
Columns	
OrganizationID (PK, int, not null)	
ParentOrganizationID (FK, int, null)	
OrganizationCodeID (FK, int, not null)	
Name (varchar(50), not null)	
Abbreviation (varchar(15), null)	
SA_ContactPersonID (FK, int, not null)	
CreatorID (FK, int, not null)	
ModifierID (FK, int, null)	
CreationDate (datetime, null)	
ModificationDate (datetime, null)	
DisplayName (varchar(50), not null)	

Polaris.OrganizationsCollections	
Columns	
OrganizationID (PK, FK, int, not null)	
CollectionID (PK, FK, int, not null)	

Polaris.Collections	
Columns	
CollectionID (PK, int, not null)	
Name (varchar(80), not null)	
Abbreviation (varchar(15), not null)	
CreatorID (FK, int, not null)	
ModifierID (FK, int, null)	
CreationDate (datetime, null)	
ModificationDate (datetime, null)	

Polaris.Organizations	
Columns	
OrganizationID (PK, int, not null)	
ParentOrganizationID (FK, int, null)	
OrganizationCodeID (FK, int, not null)	
Name (varchar(50), not null)	
Abbreviation (varchar(15), null)	
SA_ContactPersonID (FK, int, not null)	
CreatorID (FK, int, not null)	
ModifierID (FK, int, null)	
CreationDate (datetime, null)	
ModificationDate (datetime, null)	
DisplayName (varchar(50), not null)	

Polaris.OrganizationsFloatingCollections	
Columns	
HomeBranchID (PK, FK, int, not null)	
CollectionID (PK, FK, int, not null)	

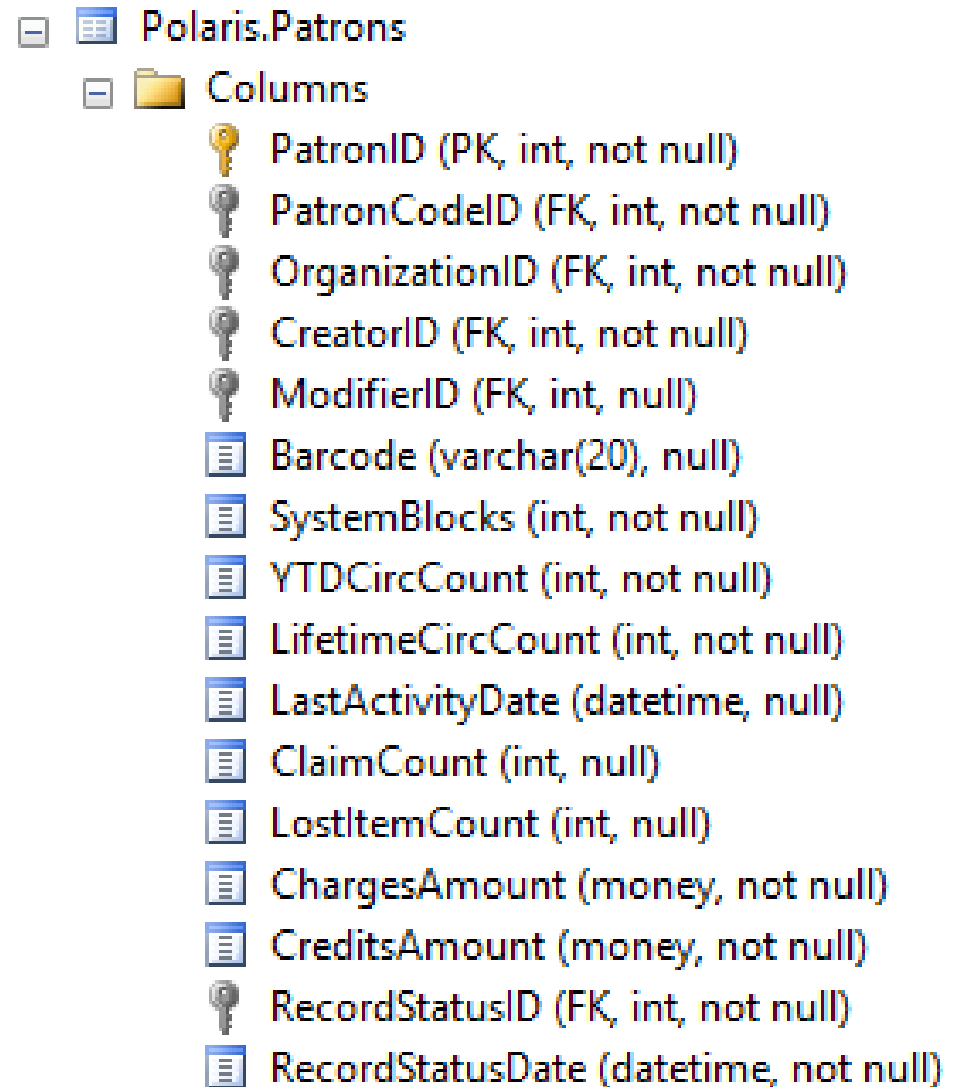
Polaris.OrganizationCodes	
Columns	
OrganizationCodeID (PK, int, not null)	
Description (varchar(80), not null)	

Polaris.SA_ContactPersons	
Columns	
SA_ContactPersonID (PK, int, not null)	
Name (varchar(50), null)	
EmailAddress (varchar(64), null)	
PhoneNumberOne (varchar(20), null)	
PhoneNumberTwo (varchar(20), null)	
FaxNumber (varchar(15), null)	

MOST COMMONLY USED TABLES

PATRONS

- Polaris.Patrons
- Polaris.PatronRegistration
- Polaris.PatronAccount
- Polaris.PatronCodes
- So many ...



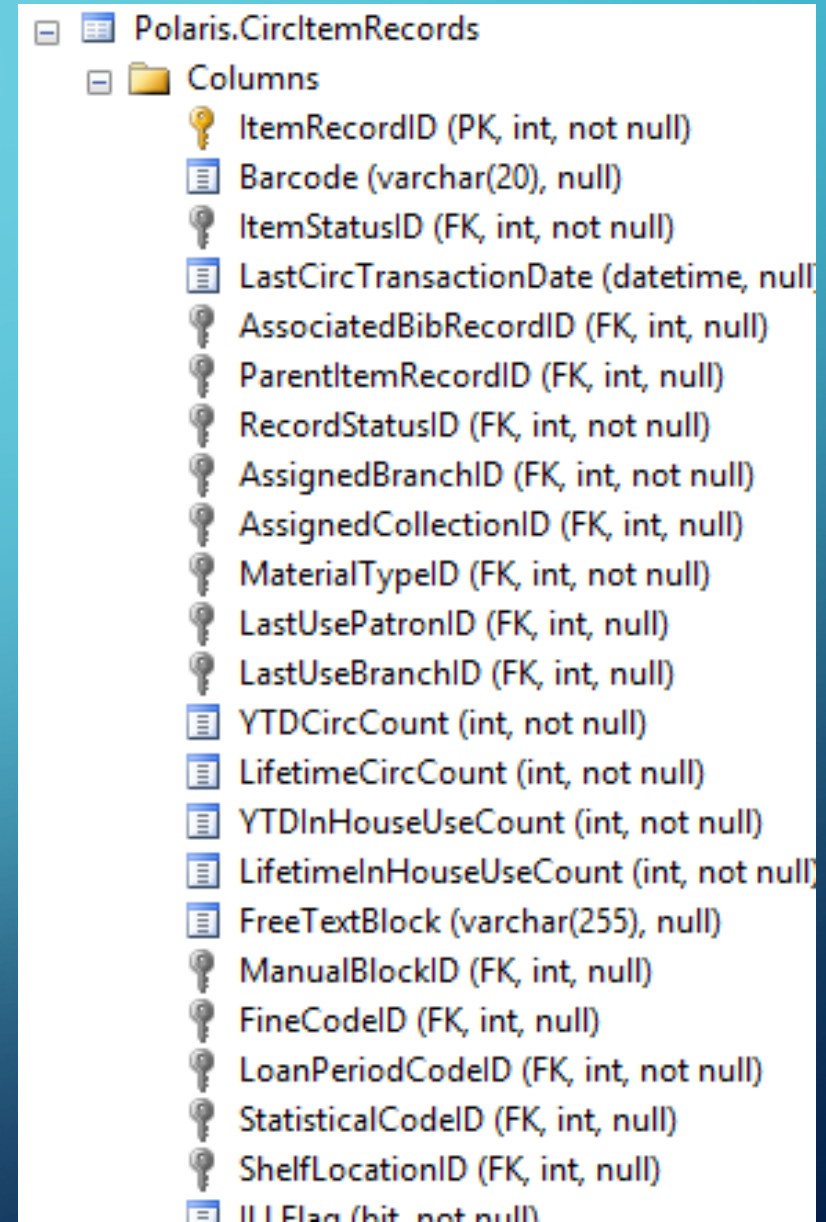
The screenshot shows a database schema viewer for the 'Polaris.Patrons' table. It lists 17 columns with their data types and constraints. The first five columns are primary or foreign keys, indicated by key icons. The remaining columns are standard data types like varchar, int, datetime, and money, indicated by list icons.

Polaris.Patrons	
Columns	
PatronID	(PK, int, not null)
PatronCodeID	(FK, int, not null)
OrganizationID	(FK, int, not null)
CreatorID	(FK, int, not null)
ModifierID	(FK, int, null)
Barcode	(varchar(20), null)
SystemBlocks	(int, not null)
YTD CircCount	(int, not null)
Lifetime CircCount	(int, not null)
LastActivityDate	(datetime, null)
ClaimCount	(int, null)
LostItemCount	(int, null)
ChargesAmount	(money, not null)
CreditsAmount	(money, not null)
RecordStatusID	(FK, int, not null)
RecordStatusDate	(datetime, not null)

MOST COMMONLY USED TABLES

ITEMS

- Polaris.CircItemRecords
- Polaris.ItemRecordDetails
- Polaris.ItemRecordHistory
- Polaris.ItemStatuses
- Polaris.MaterialTypes
- Polaris.Collections



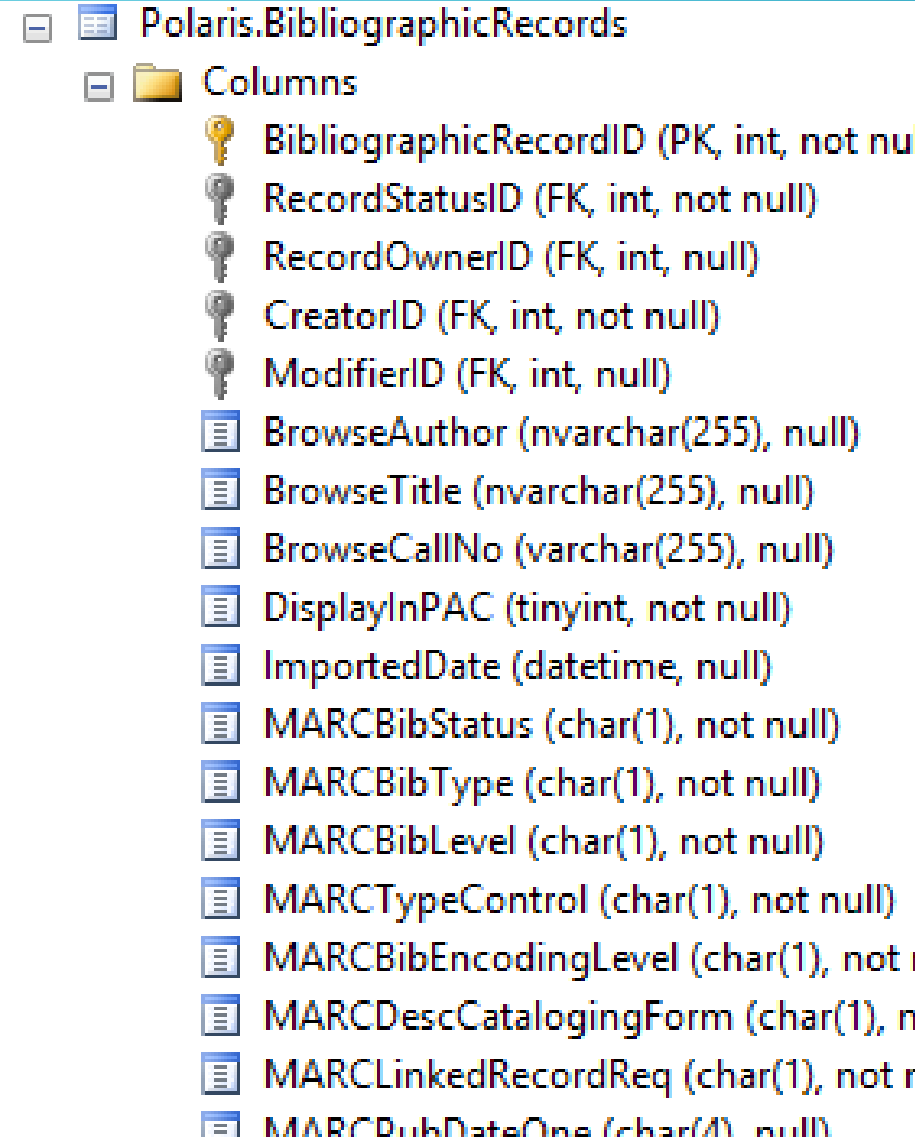
The screenshot displays the 'Polaris.CircItemRecords' table structure in a database management tool. The table is expanded to show its columns, each with a corresponding icon (key for primary/foreign keys, document for text/numeric fields) and its data type and constraints.

Column Name	Data Type	Constraints
ItemRecordID	int	PK, not null
Barcode	varchar(20)	null
ItemStatusID	int	FK, not null
LastCircTransactionDate	datetime	null
AssociatedBibRecordID	int	FK, null
ParentItemRecordID	int	FK, null
RecordStatusID	int	FK, not null
AssignedBranchID	int	FK, not null
AssignedCollectionID	int	FK, null
MaterialTypeID	int	FK, not null
LastUsePatronID	int	FK, null
LastUseBranchID	int	FK, null
YTDCircCount	int	not null
LifetimeCircCount	int	not null
YTDInHouseUseCount	int	not null
LifetimeInHouseUseCount	int	not null
FreeTextBlock	varchar(255)	null
ManualBlockID	int	FK, null
FineCodeID	int	FK, null
LoanPeriodCodeID	int	FK, not null
StatisticalCodeID	int	FK, null
ShelfLocationID	int	FK, null
ILLFlag	bit	not null



















MOST COMMONLY USED TABLES

BIBLIOGRAPHIC DATA

- Polaris.BibliographicRecords
- Polaris.BibliographicTags
 - Polaris.BibliographicTagXXXIndex



The screenshot displays the database schema for the Polaris.BibliographicRecords table. It lists 20 columns with their respective data types and constraints. The first five columns are primary or foreign keys, indicated by key icons. The remaining columns are text or date fields, indicated by document icons.

Polaris.BibliographicRecords	
Columns	
	BibliographicRecordID (PK, int, not null)
	RecordStatusID (FK, int, not null)
	RecordOwnerID (FK, int, null)
	CreatorID (FK, int, not null)
	ModifierID (FK, int, null)
	BrowseAuthor (nvarchar(255), null)
	BrowseTitle (nvarchar(255), null)
	BrowseCallNo (varchar(255), null)
	DisplayInPAC (tinyint, not null)
	ImportedDate (datetime, null)
	MARCBibStatus (char(1), not null)
	MARCBibType (char(1), not null)
	MARCBibLevel (char(1), not null)
	MARCTypeControl (char(1), not null)
	MARCBibEncodingLevel (char(1), not null)
	MARCDescCatalogingForm (char(1), not null)
	MARCLinkedRecordReq (char(1), not null)
	MARCPubDateOne (char(4), null)

MOST COMMONLY USED TABLES

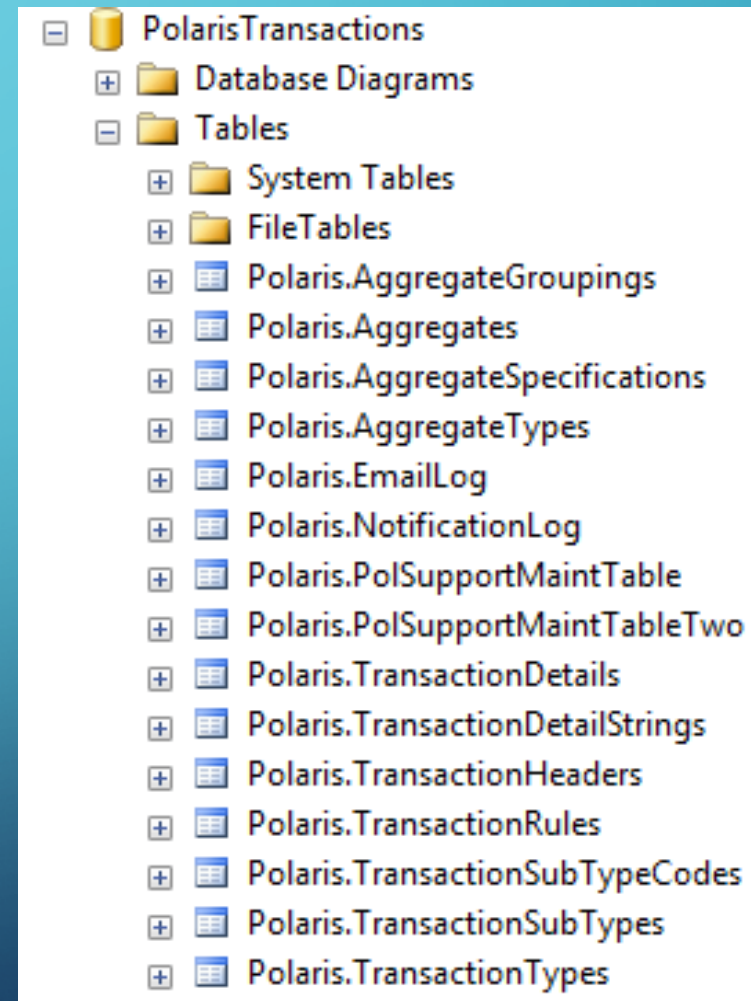
FINANCIALS

- Polaris.PurchaseOrders
 - Polaris.POLines
 - Polaris.POLineItemSegments
- Polaris.Invoices
 - Polaris.InvLines
- Polaris.Funds
 - Polaris.FundTransactionTypes
- Polaris.RWRITERAcqLinesToItemsView

+		Polaris.POAmounts
+		Polaris.PolarisCustomDataTypes
+		Polaris.PolarisDBVersion
+		Polaris.PolarisUserLists
+		Polaris.PolarisUsers
+		Polaris.POLIDuplicateSegmentLinks
+		Polaris.POLIDuplicateSegments
+		Polaris.POLineItemSegmentAmts
+		Polaris.POLineItemSegments
+		Polaris.POLineItemSegmentStatusHistories
+		Polaris.POLines
+		Polaris.POLinesStatusHistories
+		Polaris.PORelItemCreateErrors

TRANSACTION DATABASE

- Transactional Information
- Preserves Statistics
- Header Table ->
 - Details Table ->
 - Sub Types Table



TRANSACTION DATABASE

HEADER TABLES



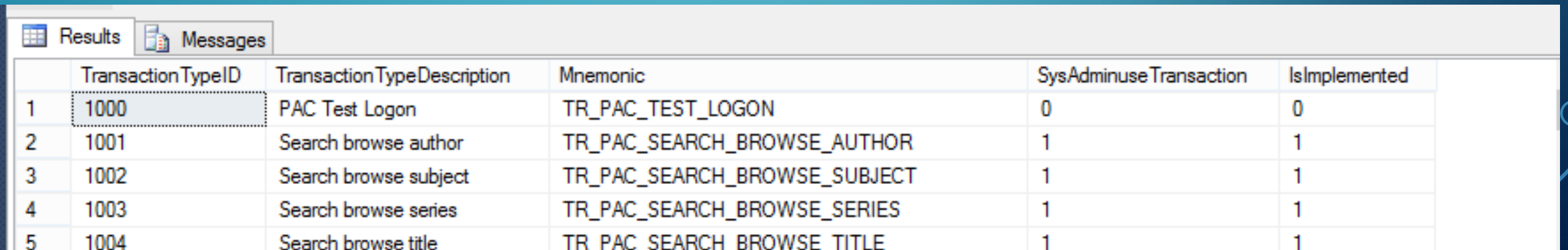
- PolarisTransactions.Polaris.TransactionHeaders
- PolarisTransactions.Polaris.TransactionTypes
- PolarisTransactions.Polaris.TransactionSubTypes

Results		Messages					
	TransactionID	OrganizationID	WorkstationID	PolarisUserID	TransactionDate	TransactionTypeID	TranClientDate
1	1	1	1	1	2008-12-09 23:12:04.080	1006	2008-12-09 23:11:59.000
2	2	1	1	1	2008-12-17 14:12:11.700	1006	2008-12-17 14:12:05.000
3	3	1	1	1	2008-12-17 14:12:15.960	1006	2008-12-17 14:12:16.000
4	4	1	1	1	2008-12-17 14:12:22.480	1006	2008-12-17 14:12:22.000

TRANSACTION DATABASE

HEADER TABLES

- PolarisTransactions.Polaris.TransactionHeaders
- PolarisTransactions.Polaris.TransactionTypes
- PolarisTransactions.Polaris.TransactionSubTypes




The screenshot shows a SQL query results window with two tabs: 'Results' and 'Messages'. The 'Results' tab is active, displaying a table with six columns: TransactionTypeID, TransactionTypeDescription, Mnemonic, SysAdminuseTransaction, and IsImplemented. The table contains five rows of data. The first row is highlighted with a dotted border.

	TransactionTypeID	TransactionTypeDescription	Mnemonic	SysAdminuseTransaction	IsImplemented
1	1000	PAC Test Logon	TR_PAC_TEST_LOGON	0	0
2	1001	Search browse author	TR_PAC_SEARCH_BROWSE_AUTHOR	1	1
3	1002	Search browse subject	TR_PAC_SEARCH_BROWSE_SUBJECT	1	1
4	1003	Search browse series	TR_PAC_SEARCH_BROWSE_SERIES	1	1
5	1004	Search browse title	TR_PAC_SEARCH_BROWSE_TITLE	1	1

TRANSACTION DATABASE

HEADER TABLES

- PolarisTransactions.Polaris.TransactionHeaders
- PolarisTransactions.Polaris.TransactionTypes
- PolarisTransactions.Polaris.TransactionSubTypes




	TransactionTypeID	TransactionTypeDescription	Mnemonic	SysAdminuseTransaction	IsImplemented
1	2011	OBSOLETE: Renewal Overdue Block	TR_PATSERV_RENEWALOVERDUEBLOCK	0	0
2	2012	OBSOLETE: Renewal Overdue Block Override	TR_PATSERV_RENEWALOVERDUEBLOCKOV	0	0
3	2080	OBSOLETE: Charge	TR_PATSERV_CHARGE	0	0
4	2081	OBSOLETE: Alter Charge	TR_PATSERV_CHARGE_ALTERED	0	0
5	2090	OBSOLETE: Add hold request	TR_PATSERV_ADDHOLDREQUEST	0	0

TRANSACTION DATABASE

HEADER TABLES

- PolarisTransactions.Polaris.TransactionHeaders
- PolarisTransactions.Polaris.TransactionTypes
- PolarisTransactions.Polaris.TransactionSubTypes



Results		Messages			
	TransactionSubTypeID	TransactionSub TypeDescription	Data Type	Mnemonic	IsImplemented
1	1	OBSOLETE: ItemDBID	K	ITEM_DB_ID	0
2	2	OBSOLETE: ItemID	K	ITEM_ID	0
3	3	OBSOLETE: FineCode	K	FINE_CODE	0
4	4	MaterialType	K	MATERIAL_TYPE	1
5	5	OBSOLETE: PatronDBID	K	PATRON_DB_ID	0



TRANSACTION DATABASE

TRANSACTION RULES

- `PolarisTransactions.Polaris.TransactionRules`
- 
- 
- 

TRANSACTION DATABASE

TRANSACTION RULES

- PolarisTransactions.Polaris.TransactionRules

```
SELECT *  
FROM Polaris.TransactionRules  
WHERE TransactionTypeID = 6001 -- checkouts
```


TRANSACTION DATABASE

TRANSACTION RULES

- PolarisTransactions.Polaris.TransactionRules

Results			Messages		
	TransactionTypeID	TransactionSubTypeID			
1	6001	4			
2	6001	6			
3	6001	7			
4	6001	33			
5	6001	38			

TRANSACTION DATABASE

TRANSACTION RULES

- PolarisTransactions.Polaris.TransactionRules

```
= SELECT TR.TransactionTypeID, TST.TransactionSubTypeID, TST.TransactionSubTypeDescription  
FROM Polaris.TransactionRules TR  
JOIN Polaris.TransactionSubTypes TST ON  
TST.TransactionSubTypeID = TR.TransactionSubTypeID  
WHERE TR.TransactionTypeID = 6001 -- checkouts|
```

TRANSACTION DATABASE

TRANSACTION RULES

Results		Messages	
	TransactionTypeID	TransactionSubTypeID	TransactionSub TypeDescription
1	6001	4	MaterialType
2	6001	6	PatronID
3	6001	7	PatronCode
4	6001	33	PatronStatClassCode
5	6001	38	ItemRecordID
6	6001	60	Item Statistical Code
7	6001	61	Assigned Collection Code
8	6001	123	Patrons Assigned Branch ID
9	6001	124	Renewal
10	6001	125	Items Assigned Branch ID
11	6001	145	Checkout Type
12	6001	186	Checkout Date
13	6001	220	CourseReserveID
14	6001	224	TermID
15	6001	233	HoldRequestID
16	6001	264	NextHoldRequestID
17	6001	268	Item Home BranchID
18	6001	300	Vendor Account ID
19	6001	301	Electronic Item

TRANSACTION DATABASE

TRANSACTION DETAILS

- PolarisTransactions.Polaris.TransactionDetails

Results		Messages		
	TransactionID	TransactionSubTypeID	numValue	dateValue
1	1	23	1	NULL
2	2	23	2	NULL
3	3	23	3	NULL
4	4	23	3	NULL
5	5	23	4	NULL
6	6	23	5	NULL

TRANSACTION DATABASE

TRANSACTION DETAILS

- PolarisTransactions.Polaris.TransactionDetails

```
= SELECT TH.TransactionID, TH.TranClientDate,  
    TD.TransactionSubTypeID, TD.numValue  
FROM Polaris.TransactionHeaders TH  
JOIN Polaris.TransactionDetails TD ON  
    TH.TransactionID = TD.TransactionID  
WHERE TH.TransactionID = 436187 -- limiting to as single transaction
```

TRANSACTION DATABASE

TRANSACTION DETAILS

- PolarisTransactions.Polaris.TransactionDetails

Results		Messages		
	TransactionID	TranClientDate	TransactionSubTypeID	numValue
1	436187	2009-01-15 12:41:06.000	4	1
2	436187	2009-01-15 12:41:06.000	6	48096
3	436187	2009-01-15 12:41:06.000	7	6
4	436187	2009-01-15 12:41:06.000	33	90097
5	436187	2009-01-15 12:41:06.000	38	2576505
6	436187	2009-01-15 12:41:06.000	60	NULL
7	436187	2009-01-15 12:41:06.000	61	140
8	436187	2009-01-15 12:41:06.000	123	177
9	436187	2009-01-15 12:41:06.000	124	1
10	436187	2009-01-15 12:41:06.000	125	177
11	436187	2009-01-15 12:41:06.000	145	15
12	436187	2009-01-15 12:41:06.000	186	NULL

TRANSACTION DATABASE

TRANSACTION DETAILS

```
= SELECT TH.TransactionID, TH.TranClientDate,  
    TD.TransactionSubTypeID, TD.numValue,  
    TST.TransactionSubTypeDescription  
FROM Polaris.TransactionHeaders TH  
JOIN Polaris.TransactionDetails TD ON  
    TH.TransactionID = TD.TransactionID  
JOIN Polaris.TransactionSubTypes TST ON  
    TST.TransactionSubTypeID = TD.TransactionSubTypeID  
WHERE TH.TransactionID = 436187 -- limiting to as single transaction|
```

TRANSACTION DATABASE

TRANSACTION DETAILS



Results



Messages

	TransactionID	TranClientDate	TransactionSubTypeID	numValue	TransactionSub TypeDescription
1	436187	2009-01-15 12:41:06.000	4	1	MaterialType
2	436187	2009-01-15 12:41:06.000	6	48096	PatronID
3	436187	2009-01-15 12:41:06.000	7	6	PatronCode
4	436187	2009-01-15 12:41:06.000	33	90097	PatronStatClassCode
5	436187	2009-01-15 12:41:06.000	38	2576505	ItemRecordID
6	436187	2009-01-15 12:41:06.000	60	NULL	Item Statistical Code
7	436187	2009-01-15 12:41:06.000	61	140	Assigned Collection Code
8	436187	2009-01-15 12:41:06.000	123	177	Patrons Assigned Branch ID
9	436187	2009-01-15 12:41:06.000	124	1	Renewal
10	436187	2009-01-15 12:41:06.000	125	177	Items Assigned Branch ID
11	436187	2009-01-15 12:41:06.000	145	15	Checkout Type
12	436187	2009-01-15 12:41:06.000	186	NULL	Checkout Date

TRANSACTION DATABASE

TRANSACTION SUB TYPE CODES

- PolarisTransactions.Polaris.TransactionSubTypeCodes

Results		Messages	
	TransactionSubTypeID	TransactionSub TypeCode	TransactionSub TypeCodeDesc
1	105	2	check
2	105	3	voucher
3	124	1	Renewal
4	128	1	Normal
5	128	2	Bulk Check in
6	128	3	Inventory
7	128	4	Renewal stopped, item Held Checkin
8	128	5	Quick Check in
9	128	6	In house check in
10	128	7	Offline Check in

TRANSACTION DATABASE

TRANSACTION DETAILS

```
SELECT TH.TransactionID, TH.TranClientDate,  
       TD.TransactionSubTypeID, TD.numValue,  
       TST.TransactionSubTypeDescription,  
       TSTC.TransactionSubTypeCodeDesc  
FROM Polaris.TransactionHeaders TH  
JOIN Polaris.TransactionDetails TD ON  
      TH.TransactionID = TD.TransactionID  
JOIN Polaris.TransactionSubTypes TST ON  
      TST.TransactionSubTypeID = TD.TransactionSubTypeID  
LEFT JOIN Polaris.TransactionSubTypeCodes TSTC ON  
      TD.TransactionSubTypeID = TSTC.TransactionSubTypeID  
      AND TD.numValue = TSTC.TransactionSubTypeCode  
WHERE TH.TransactionID = 436187 -- limiting to as single transaction
```

TRANSACTION DATABASE

TRANSACTION DETAILS

7	436187	2009-01-15 12:41:06.000	61	140	Assigned Collection Code	NULL
8	436187	2009-01-15 12:41:06.000	123	177	Patrons Assigned Branch ID	NULL
9	436187	2009-01-15 12:41:06.000	124	1	Renewal	Renewal
10	436187	2009-01-15 12:41:06.000	125	177	Items Assigned Branch ID	NULL
11	436187	2009-01-15 12:41:06.000	145	15	Checkout Type	Circ Checkout and Renewal
12	436187	2009-01-15 12:41:06.000	186	NULL	Checkout Date	NULL
13	436187	2009-01-15 12:41:06.000	220	NULL	CourseReserveID	NULL

TRANSACTION DATABASE

JOINING THE CORE DATABASE

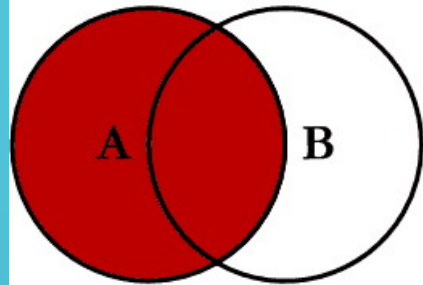
- Polaris.Polaris.Collections
- Polaris.Polaris.CircItemRecords
- Polaris.Polaris.Organizations

```
LEFT JOIN Polaris.Polaris.Collections PC ON  
TD.TransactionSubTypeID = 61 -- collection code transactionsubtype  
AND TD.numValue = PC.CollectionID  
LEFT JOIN Polaris.Polaris.CircItemRecords CIR ON  
TD.TransactionSubTypeID = 38 -- itemrecordID subtype  
AND TD.numValue = CIR.ItemRecordID  
LEFT JOIN Polaris.Polaris.Organizations PO ON  
TD.TransactionSubTypeID = 177 -- patrons assigned branch subtype  
AND TD.numValue = PO.OrganizationID
```

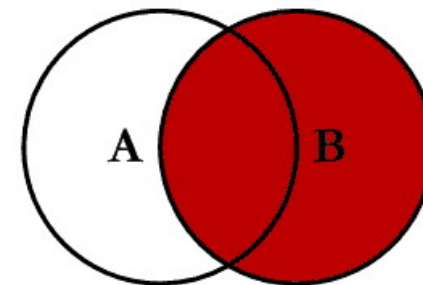

SQL BASICS

JOINS

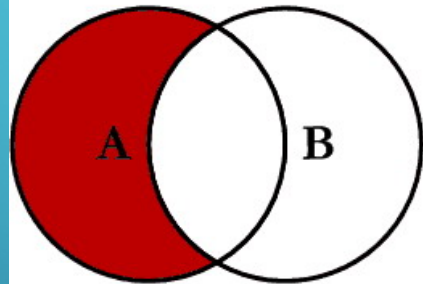
SQL JOINS



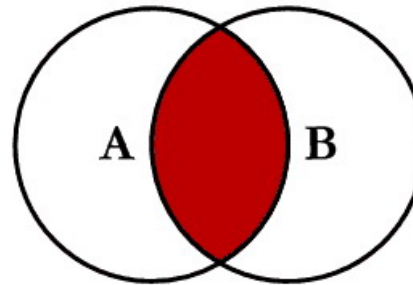
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key
```



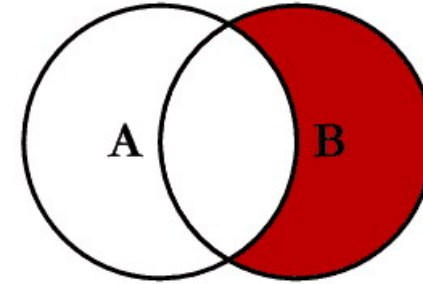
```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key
```



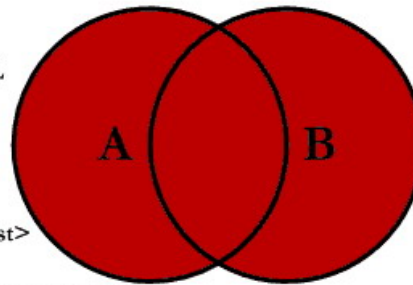
```
SELECT <select_list>  
FROM TableA A  
LEFT JOIN TableB B  
ON A.Key = B.Key  
WHERE B.Key IS NULL
```



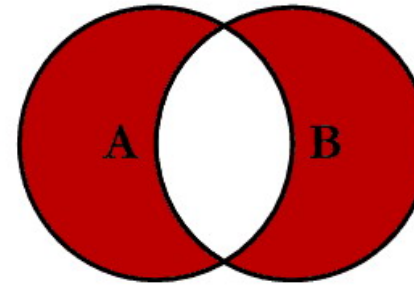
```
SELECT <select_list>  
FROM TableA A  
INNER JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
RIGHT JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key
```



```
SELECT <select_list>  
FROM TableA A  
FULL OUTER JOIN TableB B  
ON A.Key = B.Key  
WHERE A.Key IS NULL  
OR B.Key IS NULL
```

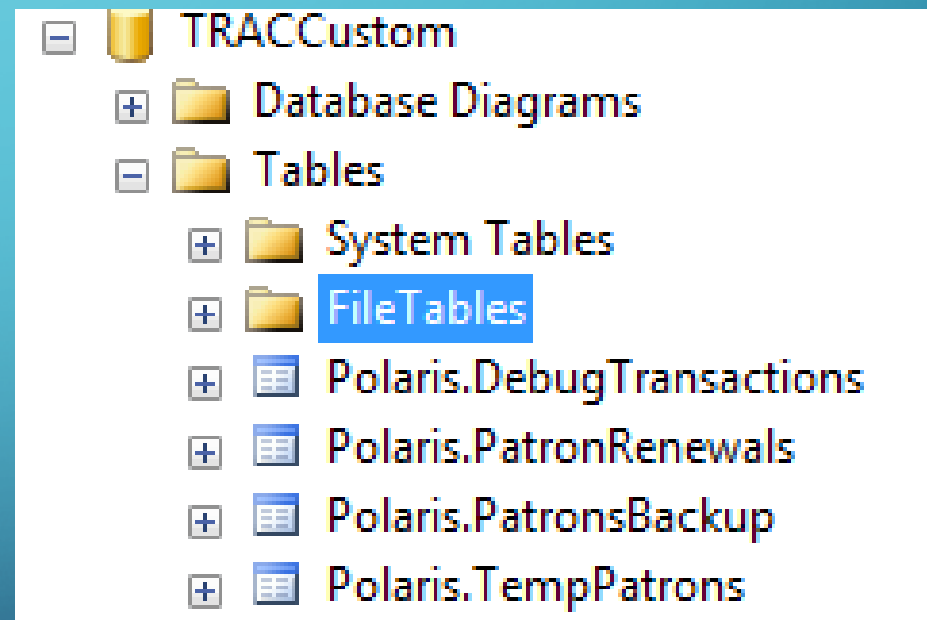
USING YOUR OWN DATABASE

- More customization
- Not lost in product upgrades
- No danger of corrupting or losing data
- Easy updates and customization

USING YOUR OWN DATABASE

DATABASE DESIGN

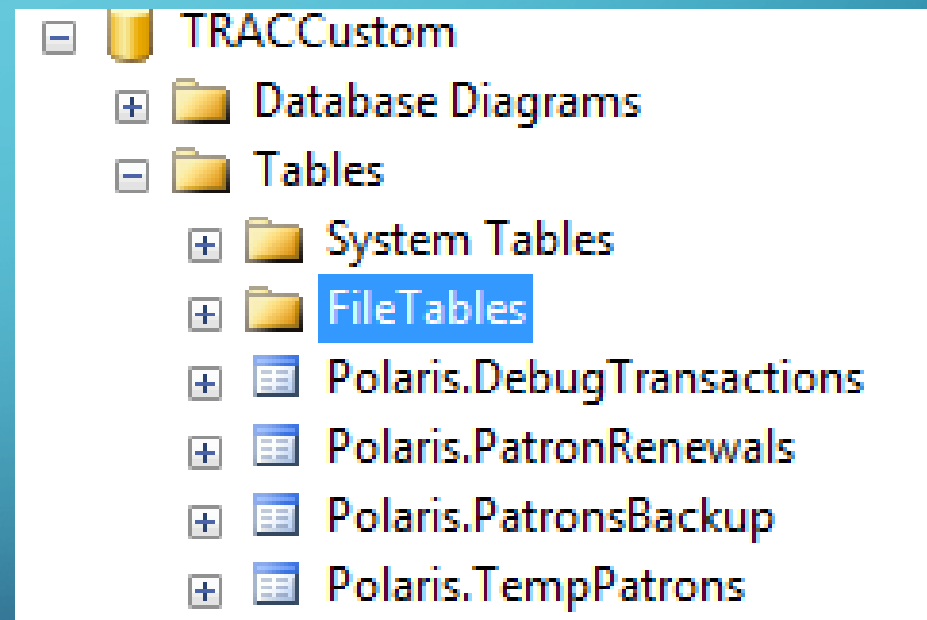
- Somewhere to track renewals
- Somewhere that stores renewal data



USING YOUR OWN DATABASE

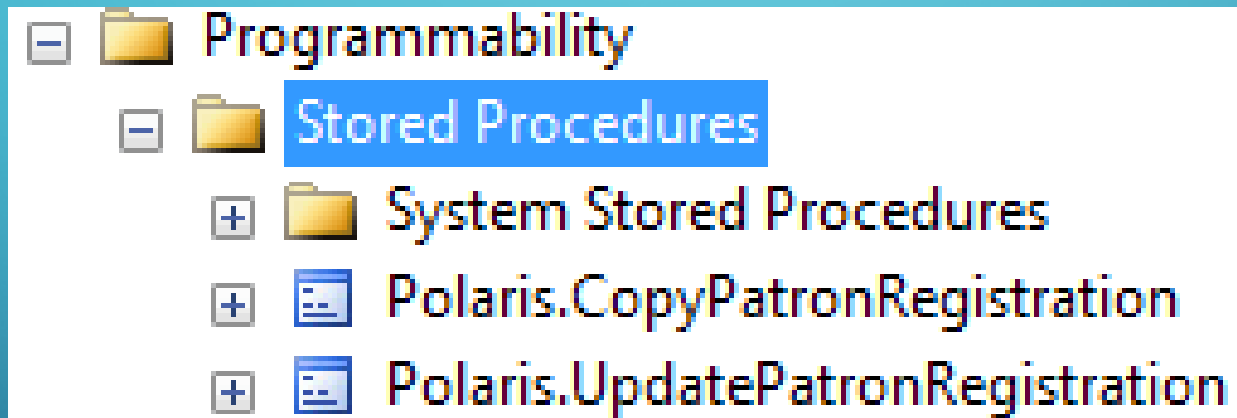
DATABASE DESIGN

- Somewhere to track renewals
- Somewhere that stores renewal data
- TRACCustom.Polaris.TempPatrons
- TRACCustom.Polaris.PatronRenewals



USING YOUR OWN DATABASE

THE RENEWAL CALCULATION



USING YOUR OWN DATABASE

THE RENEWAL CALCULATION

Job step list:

St...	Name	Type	On Success	On Failure
1	Calculated Renewals	Transact-...	Go to the...	Quit the job...
2	Copy New Information	Transact-...	Quit the j...	Quit the job...

USING YOUR OWN DATABASE

THE RENEWAL CALCULATION



PatronRegistration

The diagram consists of two light green rectangular boxes stacked vertically. Each box has a thin vertical line on its left side and a thin horizontal line at its top, creating a header-like structure. The top box is labeled 'PatronRegistration' and the bottom box is labeled 'TempPatrons'.

TempPatrons

USING YOUR OWN DATABASE

THE RENEWAL CALCULATION



PatronRegistration



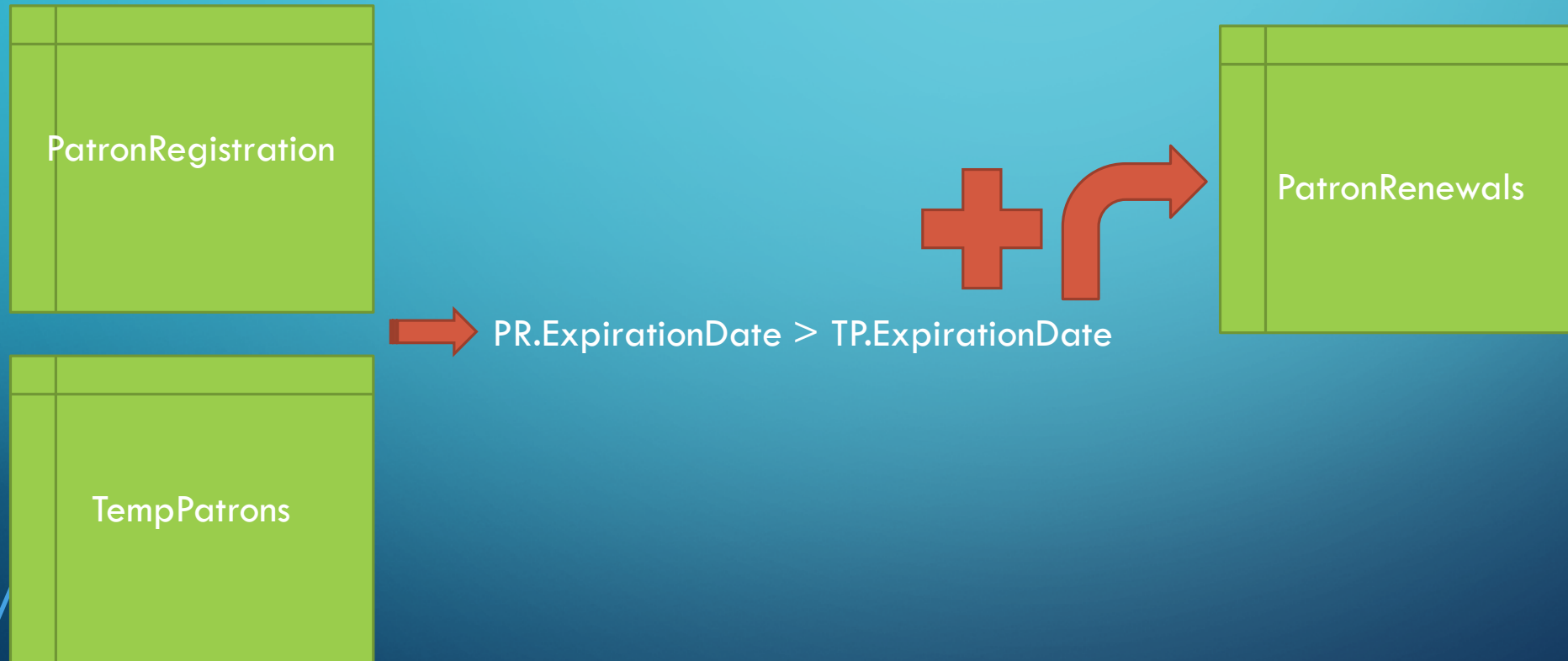
$PR.ExpirationDate > TP.ExpirationDate$



TempPatrons

USING YOUR OWN DATABASE

THE RENEWAL CALCULATION



USING YOUR OWN DATABASE

THE RENEWAL CALCULATION



BUILDING REPORTS

- How Data is displayed
 - Working with tables
 - Multiple tables in a report
 - Header Rows
 - Report Headers
- Temporary Tables

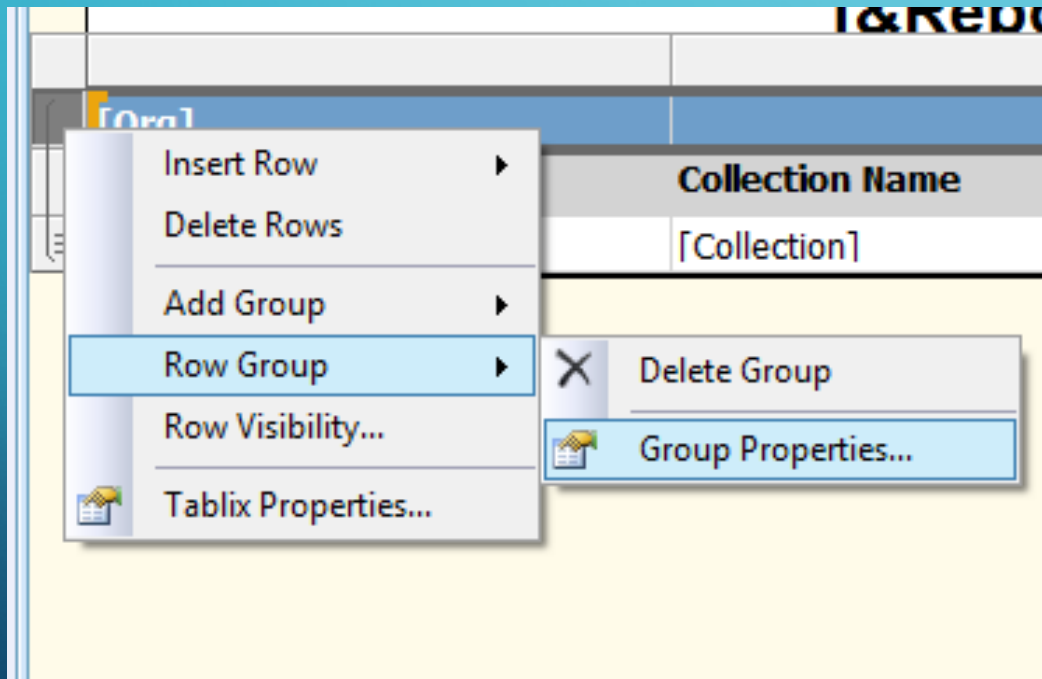
BUILDING REPORTS

USING TABLES

[&ReportName]		
[Org]	Total Items:	«Expr»
Collection Name		Items in Collection
[Collection]		[#Count]

BUILDING REPORTS

USING TABLES



Name:

Group expressions:

Group on:

REPORT EXAMPLES

USING TABLES

Collection Count By Code - MLS

Acadia Municipal Library	Total Items:	683902
Collection Name		Items in Collection
	(none)	11
	Audiobook - MP3 - Adult	17
	Audiobook - MP3 - Young Adult	1
	Young Adult	11
Berry Creek Community Library	Total Items:	683902
Collection Name		Items in Collection
	(none)	10
	Audiobook - MP3 - Adult	3
	Audiobook Cassette	5

BUILDING REPORTS

CODE BEHIND THE TABLE

```
IF @Branch = 0
BEGIN
SELECT PO.Name as 'Org',
CASE
WHEN PC.Name IS NULL THEN '(none)'
ELSE PC.Name
END as 'Collection', Count (IR.ItemRecordID) as 'Count'
FROM Polaris.ItemRecords IR
JOIN Polaris.Organizations PO
ON PO.OrganizationID = IR.OwningBranchID
LEFT JOIN Polaris.Collections PC ON
PC.CollectionID = IR.AssignedCollectionID
WHERE IR.RecordStatusID = 1 --not deleted
AND PO.ParentOrganizationID = @Parent
AND IR.MaterialTypeID NOT IN (26)
GROUP BY PO.Name, PC.Name
ORDER BY PO.Name, PC.Name
END
ELSE
BEGIN
SELECT PO.Name as 'Org',
```

BUILDING REPORTS

CODE BEHIND THE TABLE

```
SELECT '(All)' as 'Name', '0' AS OrganizationID
UNION
SELECT Name, OrganizationID
FROM Polaris.Organizations
WHERE ParentOrganizationID = @Parent
```


BUILDING REPORTS

CODE BEHIND THE TABLE

```
IF @Branch = 0
BEGIN
SELECT PO.Name as 'Org',
CASE
WHEN PC.Name IS NULL THEN '(none)'
ELSE PC.Name
END as 'Collection', Count (IR.ItemRecordID) as 'Count'
FROM Polaris.ItemRecords IR
JOIN Polaris.Organizations PO
ON PO.OrganizationID = IR.OwningBranchID
LEFT JOIN Polaris.Collections PC ON
PC.CollectionID = IR.AssignedCollectionID
WHERE IR.RecordStatusID = 1 --not deleted
AND PO.ParentOrganizationID = @Parent
AND IR.MaterialTypeID NOT IN (26)
GROUP BY PO.Name, PC.Name
ORDER BY PO.Name, PC.Name
END
ELSE
BEGIN
SELECT PO.Name as 'Org',
```

BUILDING REPORTS

MULTIPLE TABLES

[&ReportName]

«Expr»

Barcode	Name	Renewal Date
[Barcode]	[PatronFullName]	[RenewDate]
Total Renewals:		[Count(Barcode)]

Library	Renewals
[Branch_Name]	[Renewals]
Total Renewals:	[Sum(Renewals)]

«Expr»

BUILDING REPORTS

REPORT HEADERS

		[&ReportName]	
		«Expr»	
Owning Organization:		«Expr»	
Balance at Beginning of Year:		«Expr»	
Balance at Beginning of Report Period:		«Expr»	
Expended Amount:		«Expr»	
Credited Amount:		«Expr»	
Balance at end of Report Period:		«Expr»	
Total Amount for Items On Order:		«Expr»	
Free Balance:		«Expr»	
GST for the Month =		«Expr»	
		«Expr»	
		«Expr»	
Date	Title	Transaction Type	

BUILDING REPORTS

USING VARIABLES

```
/****** Header information */  
DECLARE @Fname varchar(50)  
SET @Fname = (SELECT F.Name FROM Polaris.Funds F WHERE FundID = @FundID)  
DECLARE @FALTname varchar(50)  
SET @FALTname = (SELECT F.AlternativeName FROM Polaris.Funds F WHERE FundID = @FundID)  
/* Beginning Allocation */  
DECLARE @Balloc money  
SET @Balloc = (SELECT FundTransAmt FROM Polaris.FundTransactionHistories  
                WHERE FundID = @FundID  
                AND FundTransactionTypeID = 1  
                )
```


BUILDING REPORTS


USING VARIABLES

```
SELECT
/* **** Header Information */
@Fname as 'Fund Name',
@FALTname as 'Alt Name',
@Balloc as 'Beginning Alloc',
@Bbalance as 'Prev Balance',
@EBalance as 'Encum Balance',
'GST' = (@GST * 0.05),
CONVERT(VARCHAR(10), @BDATE, 103) AS 'StartDate',
CONVERT(VARCHAR(10), @EDATE, 103) AS 'EndDate',
'PName' = (SELECT Name FROM Polaris.Organizations WHERE OrganizationID = @Parent),
/* **** */
PF.ID,
PF.Date,
substring(PF.Title,1,35) as 'Title',
'TType' =
CASE
    WHEN (PF.MyTYPE IS NULL) AND (PF.PaymentID IN (7)) THEN 'Paid'
```



BUILDING REPORTS

TABLE VARIABLES

- Can store multiple columns and rows
 - Are temporarily allocated
 - No permanent storage
 - Tables vanish at the end of the script
 - Ad-hoc easily customizable
- 

BUILDING REPORTS

TABLE VARIABLES

```
DECLARE @FinancesTEMP TABLE
(
    Date datetime,
    Title varchar(500),
    Price money,
    MyType int,
    Notes varchar(500),
    PaymentID int
)
```

```
CREATE TABLE #FinancesTEMP
(
    Date datetime,
    Title varchar(500),
    Price money,
    MyType int,
    Notes varchar(500),
    PaymentID int
)
```

BUILDING REPORTS

TABLE VARIABLES

```
INSERT INTO @FinancesTEMP (Date, Title, Price, MyType, Notes, PaymentID)
SELECT
    PILS.PaymentStatusDate as 'Date',
    PIL.Title as 'Title',
    --PILSA.Amount as 'Price',
    'Price' = (PILSA.Amount * I.ExchangeRate),
    NULL as 'Type',
    NULL as 'Notes',
    PILS.PaymentStatusID as 'PaymentID'
FROM Polaris.Funds PF WITH (NOLOCK)
```


LETS MAKE ONE!

- Time check?



The background is a blue gradient. In the corners, there are white line-art illustrations of circuit boards or neural networks, with lines and small circles representing nodes.




LET'S MAKE ONE!

DESIGN PROCESS



LETS MAKE ONE!

DESIGN PROCESS – THE DATA

- By library, what are the most popular circulated materials, and for the most popular materials, which collections are most often circulated
- 
- 
- 

LETS MAKE ONE!

DESIGN PROCESS – THE LAYOUT

Library Name	(Group 1)			
(Group 2)	Material Type		Circ volume	Percentage of Circ
(Repeat for Each)	Collection Name	Circ Volume	Percentage of Circ	

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
[- SELECT *  
  FROM PolarisTransactions.Polaris.TransactionTypes  
  WHERE IsImplemented = 1  
  ORDER BY TransactionTypeID
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
= SELECT *  
FROM PolarisTransactions.Polaris.TransactionRules TR  
JOIN PolarisTransactions.Polaris.TransactionSubTypes TST  
ON TST.TransactionSubTypeID = TR.TransactionSubTypeID  
WHERE TransactionTypeID = 6001|
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
[-] /*  
    Material type - 4  
    PatronID - 6  
    ItemRecordId - 38  
    Assigned Collectio Code - 61  
    Checkout type 145  
    Renewal 124  
*/|
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
DECLARE @BDATE datetime
DECLARE @EDATE datetime
SET @BDATE = '2015-01-01'
SET @EDATE = '2016-01-31'

SELECT *
FROM PolarisTransactions.Polaris.TransactionHeaders TH
WHERE TH.TransactionTypeID = 6001 -- checkouts only
AND TH.TranClientDate BETWEEN @BDATE AND @EDATE
```


LETS MAKE ONE!

DESIGN PROCESS – CODING

```
DECLARE @BDATE datetime
DECLARE @EDATE datetime
SET @BDATE = '2015-01-01'
SET @EDATE = '2016-01-31'

SELECT *
FROM PolarisTransactions.Polaris.TransactionHeaders TH
WHERE TH.TransactionTypeID = 6001 -- checkouts only
AND TH.TranClientDate BETWEEN @BDATE AND @EDATE
```

Polaris | 00:00:06 | 410788 rows

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
[-] DECLARE @BDATE datetime
    DECLARE @EDATE datetime
    SET @BDATE = '2015-01-01'
    SET @EDATE = '2015-01-31'

[-] CREATE TABLE #TEMPTrans (
    TransactionID int
)

[-] INSERT INTO #TEMPTrans
    SELECT TH.TransactionID
    FROM PolarisTransactions.Polaris.TransactionHeaders TH
    WHERE TH.TransactionTypeID = 6001 -- checkouts only
    AND TH.TranClientDate BETWEEN @BDATE AND @EDATE

[-] SELECT *
    FROM #TEMPTrans

    DROP TABLE #TEMPTrans|
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
SELECT *  
FROM #TEMPTrans  
WHERE TransactionID NOT IN ( SELECT TT.TransactionID FROM #TEMPTrans TT  
                             JOIN PolarisTransactions.Polaris.TransactionDetails TD  
                             ON TD.TransactionID = TT.TransactionID  
                             WHERE TD.TransactionSubTypeID = 124  
                             )
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
= SELECT TMP.TransactionID,  
  ( SELECT Col.Numvalue FROM PolarisTransactions.Polaris.TransactionDetails Col  
    WHERE Col.TransactionSubTypeID = 61  
    AND Col.TransactionID = TMP.TransactionID  
    ) as 'CollectionCode',  
  ( SELECT Mat.Numvalue FROM PolarisTransactions.Polaris.TransactionDetails Mat  
    WHERE Mat.TransactionSubTypeID = 4  
    AND Mat.TransactionID = TMP.TransactionID  
    ) as 'MaterialTypeCode'  
FROM #TEMPTrans TMP  
WHERE TMP.TransactionID NOT IN ( SELECT TT.TransactionID FROM #TEMPTrans TT  
  JOIN PolarisTransactions.Polaris.TransactionDetails TD2  
  ON TD2.TransactionID = TT.TransactionID  
  WHERE TD2.TransactionSubTypeID = 124  
  )
```


LETS MAKE ONE!

DESIGN PROCESS – CODING



Results



Messages

	TransactionID	CollectionCode	MaterialTypeCode
1	177753702	113	3
2	177753704	173	1
3	177753717	113	3
4	177753738	58	1
5	177753836	113	3

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
[-] SELECT *
```

```
FROM Polaris.Polaris.Collections
```

```
SELECT *
```

```
FROM Polaris.Polaris.MaterialTypes
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
( SELECT PC.Name FROM PolarisTransactions.Polaris.Tra
    JOIN Polaris.Polaris.Collections PC ON
    PC.CollectionID = Col.numValue
    WHERE Col.TransactionSubTypeID = 61
    AND Col.TransactionID = TMP.TransactionID
    ) as 'Collection',
( SELECT MT.Description FROM PolarisTransactions.Pola
    JOIN Polaris.Polaris.MaterialTypes MT ON
    MT.MaterialTypeID = Mat.numValue
    WHERE Mat.TransactionSubTypeID = 4
    AND Mat.TransactionID = TMP.TransactionID
    ) as 'MaterialType'
```

LETS MAKE ONE!

DESIGN PROCESS – CODING



Results




Messages

	TransactionID	Collection	MaterialType
1	177515857	Fiction - Adult	Book
2	177515877	Fiction - Adult	Book
3	177515878	Periodical	Magazine
4	177515885	Fiction - Juvenile	Book
5	177515917	Non-fiction	Book


LETS MAKE ONE!

DESIGN PROCESS – CODING




```
CREATE TABLE #TEMPTrans (  
    TransactionID int,  
    OrganizationID int  
)
```

```
INSERT INTO #TEMPTrans  
SELECT TH.TransactionID, TH.OrganizationID  
FROM PolarisTransactions.Polaris.TransactionHeaders TH  
WHERE TH.TransactionTypeID = 6001 -- checkouts only  
AND TH.TranClientDate BETWEEN @BDATE AND @EDATE
```



```
SELECT TMP.TransactionID, TMP.OrganizationID,  
( SELECT PC.Name FROM PolarisTransactions.Polaris.TransactionD  
    JOIN Polaris.Polaris.Collections PC ON  
    PC.CollectionID = Col.numValue  
    WHERE Col.TransactionSubTypeID = 61  
    AND Col.TransactionID = TMP.TransactionID  
    ) as 'Collection',
```



LETS MAKE ONE!

DESIGN PROCESS – CODING

```
= CREATE TABLE #TransStats (  
    TransactionID int,  
    OrganizaitonID int,  
    CollectionName varchar(50),  
    MatType varchar(50)  
)  
  
= INSERT INTO #TransStats  
SELECT TMP.TransactionID, TMP.OrganizationID,  
    ( SELECT PC.Name FROM PolarisTransactions.Polaris  
      JOIN Polaris.Polaris.Collections PC ON  
        PC.CollectionID = Col.numValue
```

```
WHERE TD2.TransactionSubTypeID = 124 -- Renewal Subtype  
)  
ORDER BY OrganizationID, 'MaterialType', 'Collection'|
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

Org Name	Org Circs #	Mat Type	Mat Circs	Collection	Coll Circs

LETS MAKE ONE!

DESIGN PROCESS – CODING

Org Name	Org Circs #	Mat Type	Mat Circs	Collection	Coll Circs

```
DECLARE @Results TABLE (  
    Orgname varchar(50),  
    OrgID int,  
    OrgCount int,  
    MatType varchar(50),  
    MatCount int,  
    CollName varchar(50),  
    CollCount int  
)
```


LETS MAKE ONE!

DESIGN PROCESS – CODING

```
= INSERT INTO @Results
SELECT PO.Name,
TS.OrganizationID,
NULL,
MatType,
NULL,
CollectionName,
'CollCount' = Count(*)
FROM #TransStats TS
JOIN Polaris.Polaris.Organizations PO ON
PO.OrganizationID = TS.OrganizationID
GROUP BY PO.Name, TS.OrganizationID, MatType, CollectionName
Order By PO.Name, MatType, 'CollCount' DESC
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
= UPDATE Res
  SET MatCount = ( SELECT 'MatCount' = COUNT (*)
    FROM #TransStats TS
    WHERE Res.MatType = TS.MatType
    AND Res.OrgID = TS.OrganizationID
  )
FROM @Results Res
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
UPDATE Res
SET OrgCount = ( SELECT 'OrgCount' = COUNT(*)
FROM #TransStats TS
WHERE Res.OrgID = TS.OrganizationID
)
FROM @Results Res
```

LETS MAKE ONE!

DESIGN PROCESS – CODING

```
- SELECT * FROM @Results  
ORDER BY OrgName, MatCount DESC, CollCount DESC  
  
DROP TABLE #TEMPTrans  
DROP TABLE #TransStats
```


LETS MAKE ONE!

DESIGN PROCESS – CODING

Results Messages							
	Orgname	OrgID	OrgCount	Mat Type	MatCount	CollName	CollCount
1	Acadia Municipal Library	3	117	Book	81	Picture Book	23
2	Acadia Municipal Library	3	117	Book	81	Fiction - Adult	17
3	Acadia Municipal Library	3	117	Book	81	Fiction - Juvenile	13
4	Acadia Municipal Library	3	117	Book	81	Non-fiction - Juvenile	9
5	Acadia Municipal Library	3	117	Book	81	Non-fiction	7
6	Acadia Municipal Library	3	117	Book	81	Board Book	4
7	Acadia Municipal Library	3	117	Book	81	Fiction - Young Adult	2
8	Acadia Municipal Library	3	117	Book	81	Fantasy	2
9	Acadia Municipal Library	3	117	Book	81	New Books - Fiction	1
10	Acadia Municipal Library	3	117	Book	81	International Collection	1
11	Acadia Municipal Library	3	117	Book	81	Romance Floating	1
12	Acadia Municipal Library	3	117	Book	81	Graphic Novels - Juvenile	1
13	Acadia Municipal Library	3	117	Book - Paperback	28	Paperback	27
14	Acadia Municipal Library	3	117	Book - Paperback	28	Fiction - Adult	1
15	Acadia Municipal Library	3	117	Book - Large Print	7	Large Print	6
16	Acadia Municipal Library	3	117	Book - Large Print	7	Large Print - Adult	1
17	Acadia Municipal Library	3	117	Video - DVD	1	DVD	1

DESIGN PROCESS – REPORT LAYOUT

Design the Table
Choose how to group the data in the table.

Available fields:

OrgID

Displayed fields:

Orgname
OrgCount

MatType
MatCount

CollName
CollCount

Page>

Group>

Details>

< Remove

XXXX
XXXX
XXXX
XXXX XXX XXXX
XXXX XXX XXXX
XXXX
XXXX XXX XXXX
XXXX XXX XXXX

Help < Back Next > Finish >> Cancel

LETS MAKE ONE!

DESIGN PROCESS – REPORT LAYOUT

IUG Report

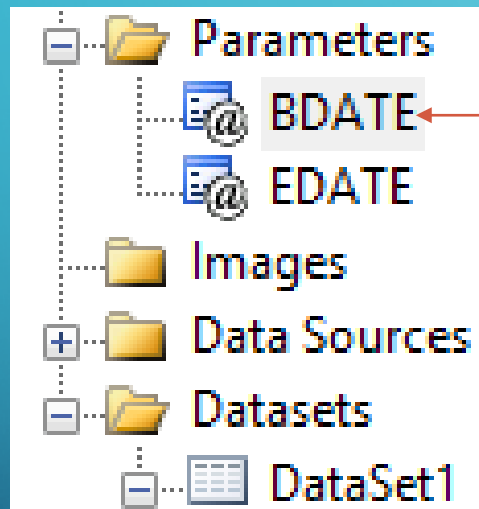
[Orgname]

[OrgCount]

Mat Tyde	Mat Count	Coll Name	Coll Count
[MatType]			
	[MatCount]		
		[CollName]	[CollCount]

LETS MAKE ONE!

DESIGN PROCESS – REPORT LAYOUT



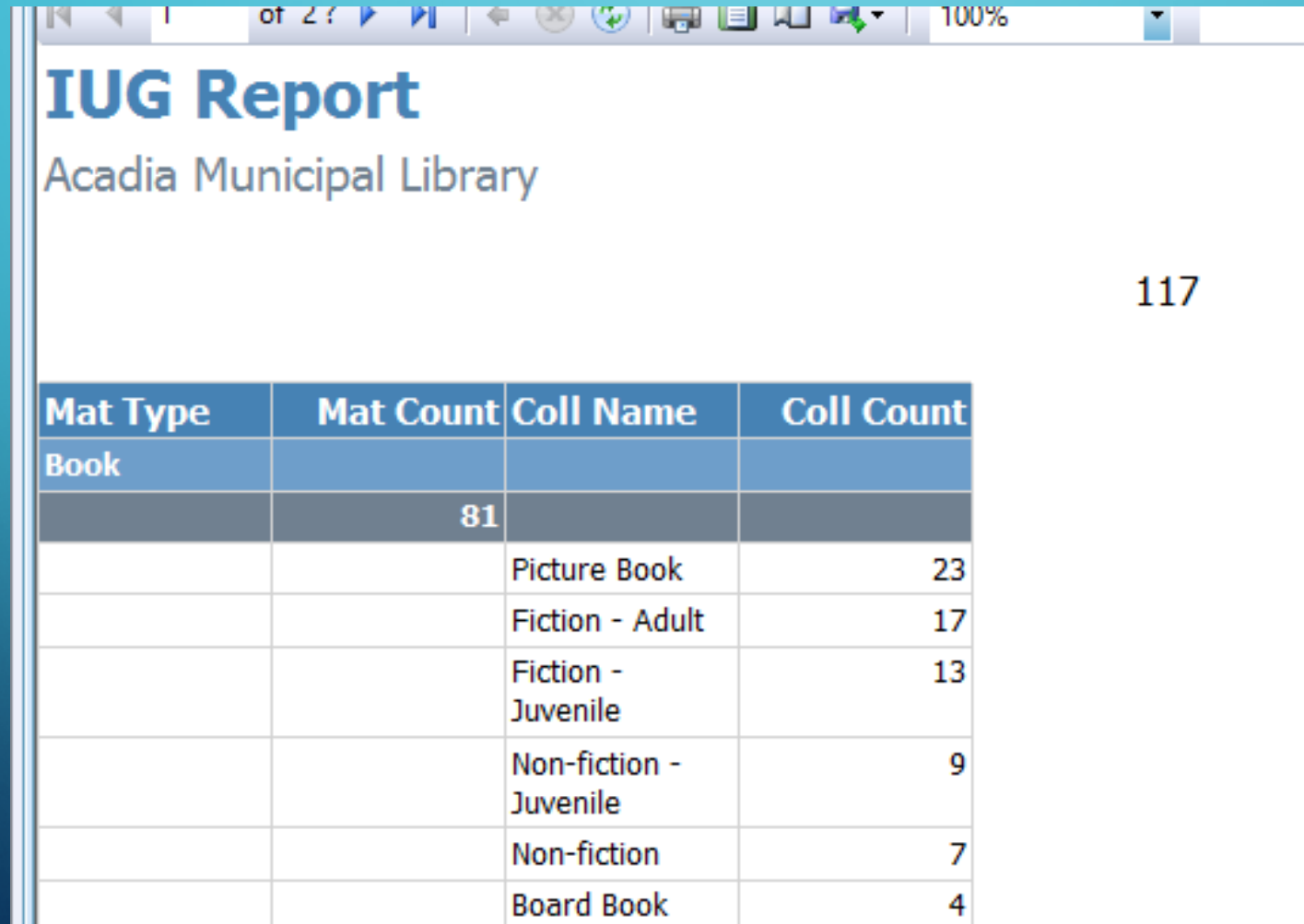
Name: BDATE

Prompt: BDATE

Data type: Date/Time

LETS MAKE ONE!

DESIGN PROCESS – REPORT LAYOUT



IUG Report
Acadia Municipal Library

117

Mat Type	Mat Count	Coll Name	Coll Count
Book			
	81		
		Picture Book	23
		Fiction - Adult	17
		Fiction - Juvenile	13
		Non-fiction - Juvenile	9
		Non-fiction	7
		Board Book	4

ALL THE ANSWERS WITH ADVANCED POLARIS SQL - THE END

PRESENTER: RICHARD KENIG

@RICHKENIG

RICHARD@MARIGOLD.AB.CA

[HTTPS://CA.LINKEDIN.COM/IN/RICHARDKENIG](https://ca.linkedin.com/in/richardkenig)

Questions

