GP Data in SQL

Understanding SQL Tables to Find GP Data

Overview of Goals

Goals of this class are:

- Overview database tables
- Where data from GP can be found
- Overview data flow in GP
- Simple SQL queries
- Joining multiple tables

Benefits

- Powerful - Can be written to meet nearly any criteria
- Used to create custom reports that meet your business needs

Overview of Data Tables

Tables

Data is stored in tables.

Similar to Excel spreadsheets

Broken into rows and columns

Columns all have unique names

Data is divided across tables with a specific dataset focus

Goal: reduce data redundancy

Normalization – every data fact should exist only once

- This simplifies updating tables and ensures the data set is more consistent - which is the major benefit of using an ERP

Each table relates to other tables based on a relation key, which can include one or multiple columns

Data in GP

- GP Module name ties to the Table Prefix

Prefix	Module
GL	General Ledger
IV	Inventory
PM	Payables Management
POP	Purchase Order Processing
RM	Receivables Management
SOP	Sales Order Processing
UPR	Payroll

First Number	Type
0	Master
1	Work
2	Open
3	History
4	Setup
5	Temporary
6	Cross Reference
7	Report Option
8	Posting Journal
9	Miscellaneous

Sample of Common Tables

Purchase Order Processing	
POP10100	PO Work Header
POP10110	PO Line Work Line
POP10300	Receipt Work Header
POP10310	Receipt Work Line

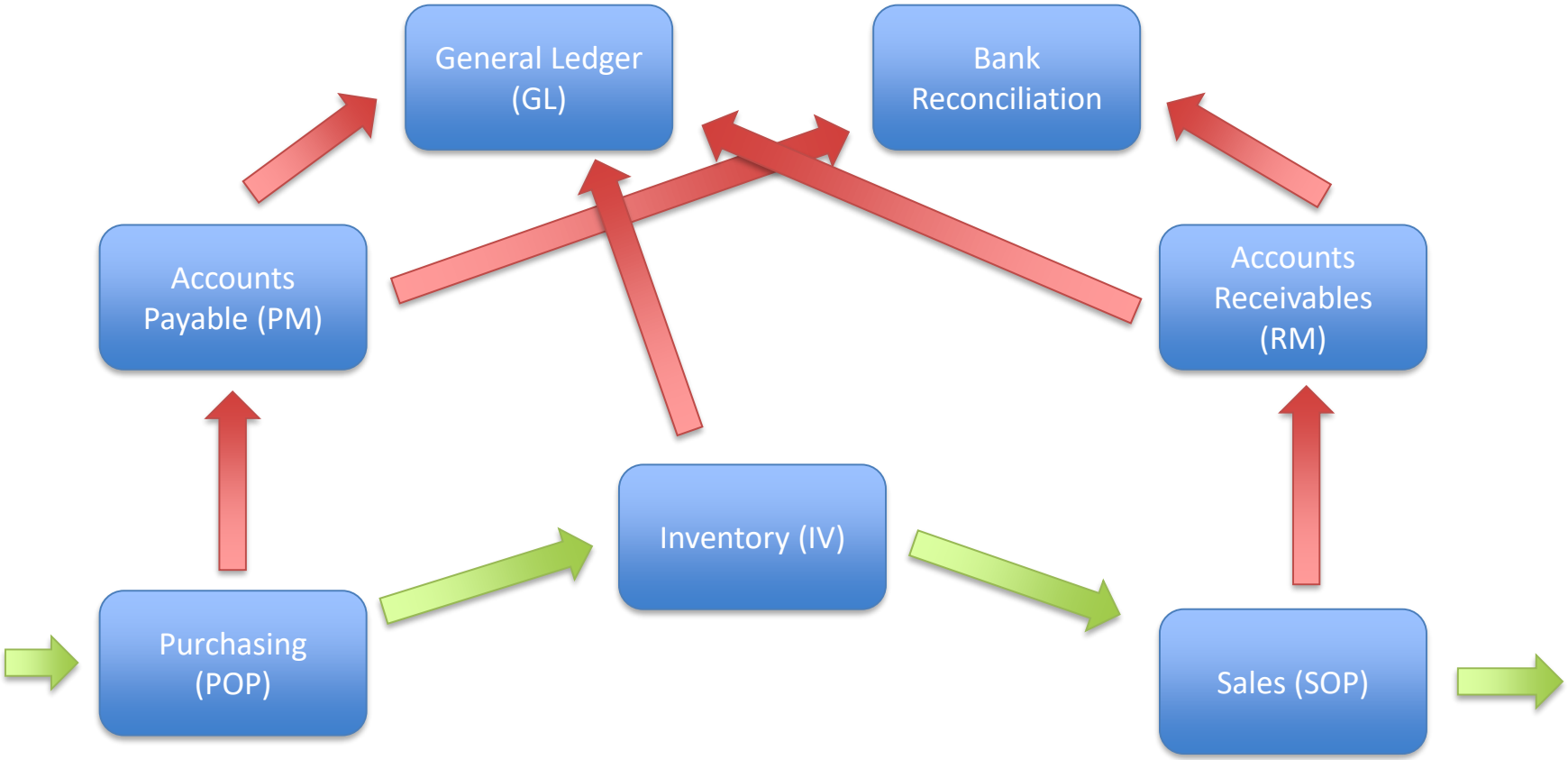
Payables Management	
PM00200	Vendor Master
PM00300	Vendor Address Master

Sales Order Processing	
SOP10100	Unposted/Work Header
SOP10200	Unposted/Work Lines
SOP30200	Historical Header
SOP30300	Historical Lines

Inventory	
IV00101	Item Master
IV00102	Item Quantity Master
IV00103	Item Vendor Master
IV00108	Item Price List
IV40400	Item Class Setup

Receivables Management	
RM00101	Customer Master
RM00102	Customer Addresses
RM00201	Customer Class Master
RM00301	Salesperson Master
RM00303	Sales Territory Master

GP Data Flow



Getting Data from SQL

Components of a simple query

select [column names] f**rom** [table name] *Where [conditional statement] Order By [columns]*

- Column names can be specified individually, or use an * to include all.
- Acronyms can be used on the table name in case we need to reference any columns in that table – important for joins. [table name] as ACRONYM
- Order By orders the data that is returned

Where Operators

Operators	Sample
=	SELECT * FROM RM00101 WHERE CUSTNMBR = 'AAR001'
!= or <>	SELECT * FROM RM00101 as RM1 WHERE RM1.CUSTCLAS != 'RETAIL'
>, <, >=, <=	SELECT * FROM RM00101 WHERE CRLMTAT > 5000
BETWEEN AND	SELECT * FROM RM00101 WHERE CRLMTAT BETWEEN 2000 AND 3000
LIKE	SELECT * FROM RM00101 WHERE CUSTNAME LIKE '%AARON00%'
IN	SELECT * FROM RM00101 WHERE STATE IN ('CT', 'MA', 'RI')

Sample Output and SQL Tricks

- Sample output

```
select * from RM00101 where custnmbr = 'AARONFIT0001'
```

	CUSTNMBR	CUSTNAME	CUSTCLAS	CPRCSTNM	CNTCPRSN	STMTNAME	SHRTNAME	ADRSCODE
1	AARONFIT0001	Aaron Fitz Electrical	USA-ILMO-T1		Bob Fitz	Aaron Fitz Electrical	Aaron Fitz Elec	PRIMARY

- Column names are abbreviated and can be difficult to read/remember

- To verify the Columns and the data in a table, use the following query:

```
select top 1 * from RM00101
```

  This will return 1 row of data, the column names and data can be verified

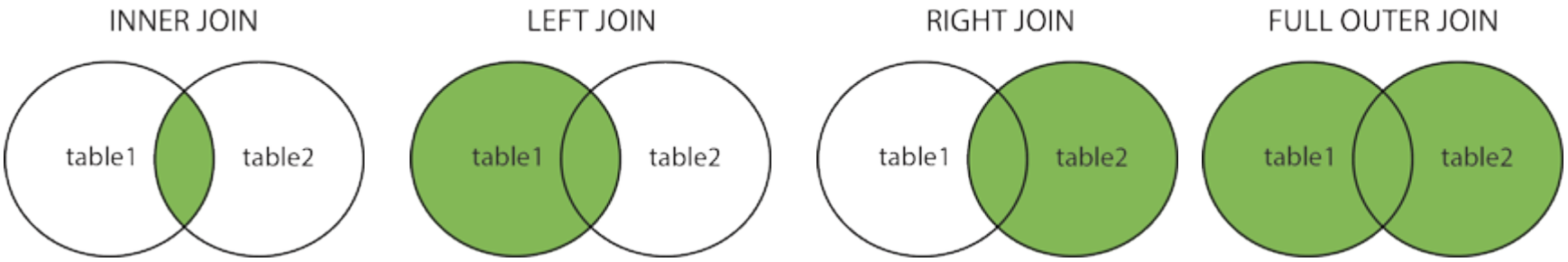
- To get a count of rows that will be returned from any query use:

```
select count(*) from RM00101
```

- Like Statements can be resource intensive and should be used with caution

Join Types

Joins are used to bring data from one or more tables together in a query. The key columns must be specified in the join. The most common join is default INNER JOIN



Example: Combining the Customer Card with the Customer Address

```
select * from RM00101 cust where CUSTNMBR = 'AARONFIT0001'
select * from RM00102 custaddr where CUSTNMBR = 'AARONFIT0001'

select * from RM00101 cust
join RM00102 custaddr on cust.CUSTNMBR = custaddr.CUSTNMBR
where cust.CUSTNMBR = 'AARONFIT0001'
```

SQL Examples

Join a second table and specify which columns to return

```
select cust.CUSTNAME, cust.CUSTNAME, cust.CUSTCLAS,
custclass.CRLMTAMT, custaddr.ADRSCODE
from RM00101 cust
join RM00102 custaddr  on cust.CUSTNMBR = custaddr.CUSTNMBR
join RM00201 custclass on cust.CUSTCLAS = custclass.CLASSID
where cust.CUSTNMBR = 'AARONFIT0001‘
```

Join two tables with multiple table keys

```
select sd.SOPNUMBE, sd.SOPTYPE, sd.CUSTNMBR, sd.CUSTNAME,
sl.ITEMNMBR, sl.ITEMDESC
from sop10100 sd
join SOP10200 sl
            on sd.SOPNUMBE = sl.SOPNUMBE
            and sd.SOPTYPE = sl.SOPTYPE
where ITEMNMBR = 'hd-20'
            and sd.SOPTYPE in (2,3)
```

Views

- Views are select statements that have been written previously which bring in data from one or more tables.
- Can be queried the same as a table
- More user friendly column names than tables
- Below are GP Views, SalesPad has many more (555)

Financial Views	Sales Views	Customer View
Accounts	SalesDistributions	CustomerAddress
AccountSummary	SalesLineItems	CustomerItems
AccountTransactions	SalesSerialLot	Customers
BankTransactions	SalesTransactions	
Purchasing Views	**Receivables Views**	**Vendor Views**
PurchaseLineItems	ReceivableApply_Open	VendorAddress
PurchaseOrders	ReceivablesTransactions	VendorItems
PurchaseOrderStatus		Vendors

Query from a view

- Example of select from a GP view:

```
select * from SalesLineItems where [item number] = 'hd-20' and [SOP Type] in ('Order', 'Invoice')
```

	SOP Type	SOP Number	Item Number	Item Description	QTY	Extended Cost	Extended Price	Unit Cost	Unit Price
1	Order	ORDST2225	HD-20	20 Gig Hard Drive	1.00000	50.00000	50.00000	50.00000	50.00000
2	Invoice	STDINV2265	HD-20	20 Gig Hard Drive	1.00000	50.00000	50.00000	50.00000	50.00000
3	Invoice	STDINV2266	HD-20	20 Gig Hard Drive	1.00000	50.00000	50.00000	50.00000	50.00000
4	Invoice	STDINV2267	HD-20	20 Gig Hard Drive	1.00000	50.00000	50.00000	50.00000	50.00000

- Example of select from a SalesPad view:

```
select * from spvSalesLineItemSearch where Item_Number = 'hd-20' and Sales_Doc_Type in ('Order', 'Invoice')
```

	Sales_Doc_Type	Sales_Doc_Num	Sales_Doc_ID	Sales_Batch	Line_Num	Component_Seq_Num	Source	Item_Number	Item_Description
1	ORDER	ORDST2231	STDORD	RDY TO INV	32768	0	History	HD-20	20 Gig Hard Drive
2	ORDER	ORDST2232	STDORD	EMAIL	32768	0	Open	HD-20	20 Gig Hard Drive
3	ORDER	ORDST2225	STDORD	NEW ORD	32768	0	Open	HD-20	20 Gig Hard Drive
4	INVOICE	ORDST2231.10	STDINV	RDY TO INV	81920	0	Open	HD-20	20 Gig Hard Drive

Aggregate Functions

- Aggregate Functions can be used to get summarized data, however they require that columns not part of the aggregate be part of a group by clause
- Aggregate functions include: MAX(), MIN(), AVG(), SUM(), COUNT()

```
select sdh.custnmbr,
            slh.itemnmbr,
            total_sold = SUM(slh.quantity)
from sop30200 sdh
            join sop30300 slh on sdh.soptype = slh.soptype
             and sdh.sopnumbe = slh.sopnumbe
where sdh.soptype = 3 and sdh.voidstts = 0
            and sdh.docdate between '01-01-2017' and '12-31-2017'
group by sdh.custnmbr, slh.itemnmbr
order by sdh.custnmbr
```

Additional Resources

Additional Resources:

- TSQL Basics
  - Microsoft Virtual Academy – SQL Database Fundamentals
  - https://mva.microsoft.com/en-US/training-courses/sql-database-fundamentals-16944
  - Microsoft Virtual Academy – Querying with Transact-SQL
  - https://mva.microsoft.com/en-US/training-courses/querying-with-transactsql-10530
- GP Table Reference
  - http://dyndeveloper.com/DynModule.aspx
  - https://victoriayudin.com/gp-tables/