Broadridge Financial Solutions Vendor Database

Final Project

IST 659

Amanda Austin

**Table of Contents**

1. Broadridge Environment
2. Broadridge Mailing Procedures and Business Rules
3. Primary Data Questions for the Database to Address
4. Logical Model
5. Creating SQL Tables
6. Connecting Access to Vendor DB using ODBC
7. Importing Historical Data
8. Inserting Movelogs into the Database
9. Assigning a Movelog to a Consolidated Skid, Load, and Vendor
10. Proof of Mailing for Client Services
11. Outbound Reporting for Invoice Approval
12. Average Processing Time
13. User Permissions
14. Summary
15. Appendix
16. **Broadridge Environment**

The investor communications division of Broadridge Financial Solutions is a large mail service provider that sends millions of pieces of mail, per week, out of the Edgewood, NY facility alone. The primary products at the Edgewood location are proxy and transaction reporting (TR) mailings to shareholders of Broadridge clients all around the world.

While the Edgewood facility mostly handles larger jobs and can send these directly to the USPS for reduced rates without additional labor, there are instances when third-party vendors allow for cost savings and efficiency. When a job cannot reach the ideal postal qualifications, it is sent to a third-party vendor, called a presort vendor, to be comingled with mail from other mail service providers. This comingling allows the vendor to receive reduced postal rates with the USPS, which are then passed onto Broadridge.

The Director of Logistics at Broadridge is looking to better manage and store their vendor outbound data. This data will then be used by the Reconciliation Team (Rec Team) to reconcile against vendor billing data and provide proof of mailing to client services. In addition to post mailing purposes, the logistic clerks need to properly input each skid of mail that is presented to logistics, then assign it to a load and determine which presort vendor the load will be sent to.

The database will focus on domestic, proxy mailings. International mail and TR product lines will be excluded – as they will need a separate database to be created in the future. International mail has additional regulations and requirements and needs to be sent to a different group of vendors. The reason TR product lines are excluded is because TR jobs are set up by different teams at Broadridge than proxy mailings. Proxy and TR jobs do not follow the same set of rules and have different regulations surrounding each. Going forward, “mail” and “jobs” will be considered domestic and proxy. In addition, Broadridge tracks mail at the skid or pallet level, not the mail piece level. Note that the words “pallet” and “skid” represent the same thing, and are used interchangeably going forward.

1. **Broadridge Mailing Procedures and Business Rules**

When a client at Broadridge requests a mailing the client representative sets up a job, which is then assigned a unique job number. A client can have zero or many jobs at Broadridge and a job is associated with exactly one client. Each client is assigned one client rep at Broadridge that the logistics team can contact if the need arises during the mailing process. The database will store the client representative’s name, email address, and phone number.

Many jobs at Broadridge are too large to fit on one pallet. For this reason, a piece of paper, called a movelog, accommodates each pallet as it moves through the warehouse. The move log is created while the mail is printing, then follows the skid as it is inserted. Eventually, it will arrive in Logistics. Each movelog contains the job number, mail class (first class, standard, or meter), piece weight, number of pieces on the skid, and a barcode that uniquely identifies the movelog. A barcode can be scanned into any computer and the unique information will be populated as if an individual typed it manually. The job number, weight, and mail class are the same for every piece of mail on one skid, and a job can have more than one move log.

All the information on the movelog is necessary for the logistic clerks to know which vendor the mail needs to go to, as well as ensuring the vendors bill Broadridge correctly. Only one employee can scan a movelog into the database, but an employee can scan many movelogs. Logistics needs to know the name of the employee that scanned into the database and their shift, as well as the day and time the movelog was scanned. Not all employees scan into the database and each employee is either on day or night shift, but not both.

There are instances when a pallet is presented to logistics with a small amount of mail on it and does not use the entire space. In order to better utilize the space on trucks and trailers the logistic clerks consolidate these smaller pallets. As in, the clerks combine smaller pallets of the same mail class and similar weight into one. Paper movelogs from each original pallet are then placed on top of the consolidated skid and travel with the skid to the vendor. Multiple move logs can be assigned to a consolidated skid however, a movelog cannot be separated into different consolidated skids. If the movelog has enough pieces and does not need to be consolidated, a consolidated skid is still created in the database and only contains the one movelog. The database will record the day and time a consolidated skid was created.

As skids are consolidated, they are loaded onto a truck or trailer. A truck or trailer is represented as a load in the database and contains multiple consolidated skids. Consolidated skids are assigned to one and only one load, as it would be inefficient to break up consolidated pallets. For each load, it is important to note the truck or trailer number, which can be found spray painted on both inside and outside of the vehicle. This truck or trailer number is not unique, as the physical truck and trailer can be used multiple times. The database will record the day and time a load was created.

Once the truck or trailer is filled, it is sent to exactly one vendor based on the mail class and mail piece weight. A presort vendor receives multiple loads of mail throughout the day. If a vendor is new to Broadridge, then it is possible that they are not associated with any loads. Each vendor has multiple vendor reps the team must contact before sending out a load so their facility can clear a bay. The database will store each vendor reps name, email, and phone number.

After the vendor has processed the mail and tendered it to the USPS, a vendor representative will send billing data to the reconciliation team. This billing data will need to be imported into the database for reconciliation purposes. As in, the reconciliation team will need to ensure the vendor is billing Broadridge for the correct number of pieces. Each instance of billing data is associated with one and only one vendor, and each vendor sends many billing files throughout the week. The billing files contain mail date, mail class, job number, piece weight, pieces, and billing rate. For ease of transition into the vendor database environment, the reconciliation team has already gone through the efforts of ensuring all billing data is consistent from vendor to vendor. In addition, each instance of billing data is associated with one and only one job, but a job can be associated with one or more billing files. This may happen if the job was sent to multiple presort vendors, the job was so large it ran over multiple dates, etc. To differentiate between the vendor information and Broadridge information, the terms “billed” and “outbound” will be used. For example, outbound pieces represent Broadridge records, while billed pieces represents the number of pieces according to the vendor.

1. **Primary Data Questions for the Database to Address**

Client service representatives may reach out to the logistics team to request proof of mailing for a specific set of jobs. The rec team will need the ability to search for these jobs in the database and know what vendor it was sent to, the mail date, and the mail class.

The rec team needs to reconcile invoices from vendors, as in, they need to ensure the vendors are billing Broadridge for the correct number of pieces. A report is needed for each vendor and mail class, that returns the number of pieces and piece weight by mail date, also referred to as outbound data.

The director and supervisors would like to know the average time it takes for a pallet of mail to leave the facility once it reaches logistics.

1. **Logical Model**

The systems department at Broadridge requires a Systems Enhancement Request (SER) and Project Definition Document (PDD) to be submitted before a project can be assigned and completed. A SER and PDD are technical documents that clearly and specifically define what needs to be completed. The logical model and below instructions will aid in these requests.

The Logical Model (Appendix 1 – Logical Model) has the necessary foreign keys (FK) to implement into SQL. All primary keys will be surrogate keys, and all date and time attributes will be assigned the data type datetime. All attributes, unless otherwise stated, are required. All composite name attributes (vendor rep name, client rep name, and employee name) will be broken out into first name and last name and assigned the varchar(30) data type.

Beginning with the vendor table, the data type varchar(50) is assigned to vendor name to allow for sufficient amount of characters if needed. Since there are multiple vendor reps per vendor but each vendor rep is only assigned to one vendor, a new entity has been created called VendorRep. The data type varchar is assigned to each attribute in vendor rep to allow for enough characters, phone number is also a varchar since no calculations will be performed on this attribute. A vendor rep must be assigned to a vendor, so the FK VendorID is required. The same logic applies to the client rep and employee entities – varchar data types are assigned to name, email and phone number. In the employee entity, shift can only be day or night so the datatype varchar(5) is assigned.

In the load table, the truck or trailer number will always be 5 characters, so a char(5) is assigned. The barcode on a movelog is unique and requires a datatype of varchar(50) to allow for sufficient characters. Outbound pieces are always whole numbers so int is assigned. Outbound piece weight will never exceed 99 ounces and only the first decimal place is needed, so decimal(3,1) is assigned. Mail class will be assigned datatype varchar(20).

For consistency, all the fields of the billing table follow the same data types as above (mail date, mail class, billed piece weight, billed pieces). Rate will be represented as a decimal, with 2 places before and after the decimal. Note: The mention of billing data up to this point is to demonstrate the entire life cycle of a job within Logistics, and the potential for the entirety of this life cycle to be contained within the vendor database. However, to keep Broadridge information and contracted rates confidential, billing will not be implemented in the physical database. Going forward, any mention of billing data or invoices will refer to Excel files or financial statements outside of the vendor database.

1. **Creating SQL Tables**

Image 1 demonstrates the creation of two tables in the vendor database. The movelog table is used as an example since it is the most involved, and consolidated skid is shown to demonstrate the relationship between the primary key and foreign key ConSkidID. The consolidated skid table needs to be created before the move log table, so the foreign key has an established primary key to reference. The creation of all other tables can be found within the SQL code (Appendix 2 – SQL Code with Result Sets).

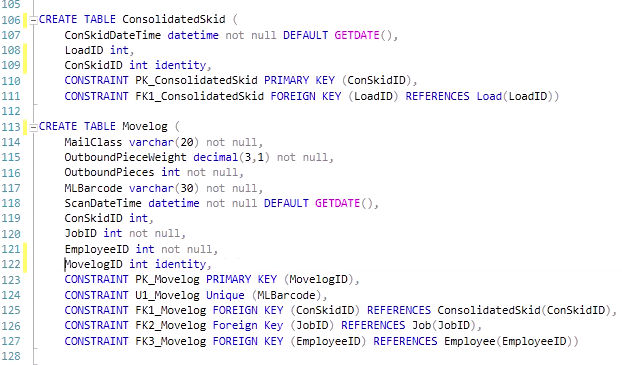


Image 1 – Creating Vendor DB Tables

1. **Connecting Access to Vendor Database Using ODBC**

Now that the tables have been created, the database will be tested in a controlled environment. Historical data will be bulk inserted into the tables through Access. Access will also serve as the user interface once the database goes live. Before Access can be utilized to import the historical data, a linked connection must be established, as demonstrated in Images 2 (below). In addition, table relationships must be established within Access to reflect the logical model (Image 3).

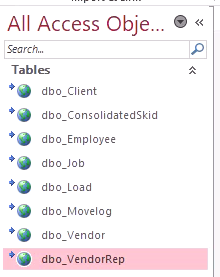


Image 2 – Establishing Link Between Access and SQL

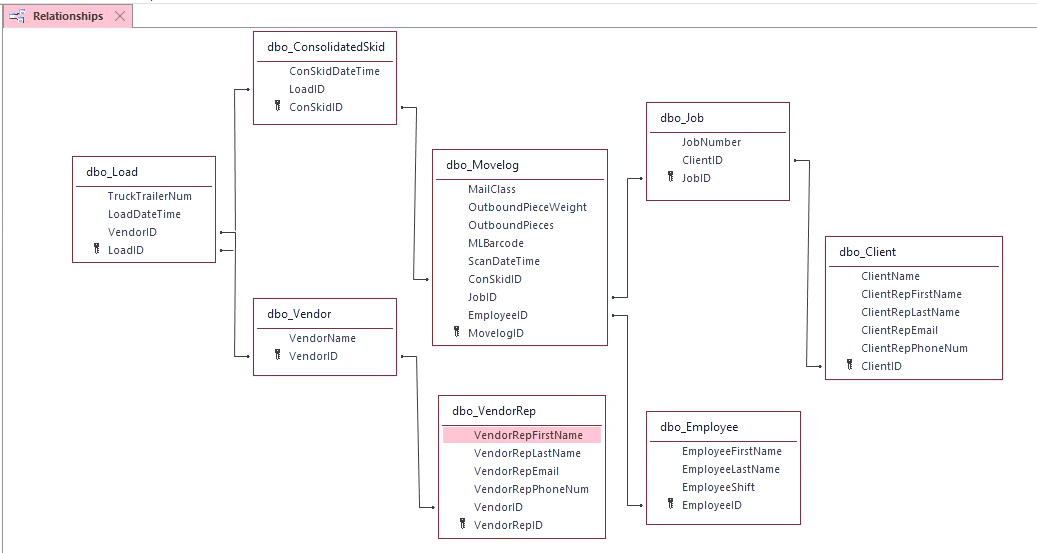


Image 3 – Establishing Relationship Between Linked Tables in Access

1. **Importing Historical Data**

All tables have been implemented in Access and are linked to the SQL data source; any changes made in Access will also be applied to the SQL server. This is desired for when the database goes live, as Access will be the primary interface. For now, Access will be used to bulk insert the historical data and to design the user interface.

Using SQL, it can be verified that the historical data was imported correctly. The movelog table can be found in Image 4 (below), the remaining tables can be found in the appendix (Appendix Images 2-8).

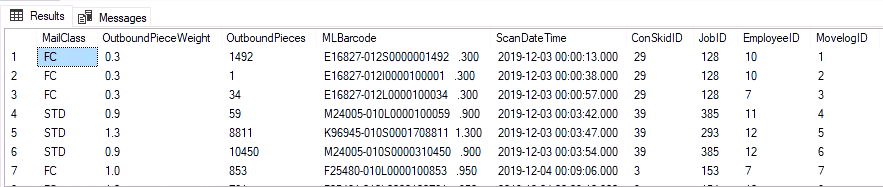


Image 4 – Movelog Data in SQL (not all data is shown)

1. **Inserting Movelogs into the Database**

The logistic clerks will use a form in Access to enter movelog information into the database. Referencing Image 5 – Scan a Movelog Form (below), the clerk will open the from and click the “Scan a new movelog” button to clear all text boxes.

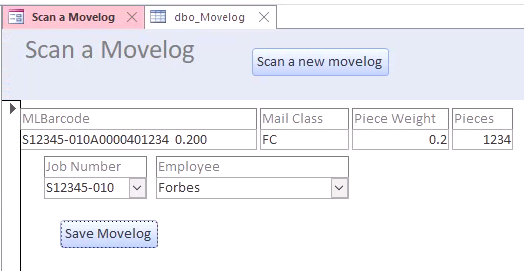


Image 5 – Scan a Movelog Form

The clerk will then scan the movelog barcode into the first field and manually enter mail class, piece weight, pieces, and job number into the remaining fields. The mail class can be found on the movelog itself, but not in the barcode. The logistic clerk will need to verify the mail piece weight and total number of pieces on the pallet before entering, as these values may be different than what is in the barcode. This is because the movelog, as mentioned earlier, is generate while the job is printing. As the mail moves from print, to inserting, and eventually to logistics, there could be damaged pieces that are no longer mailable or the piece weight could be different than predicted. In either case, the logistic clerk may only adjust the piece weight and pieces fields, and not the MLBarcode field, in order to maintain accurate records.

The logistic clerk will also pick the job number from the dropdown. All job numbers will be loaded into the vendor database when they are initially set up by the client rep. The vendor database will make use of an already existing Excel file within Operations, which contains all jobs set up but not yet processed and their respective clients. This file will be bulk inserted into the Vendor Database two times a day, once during night shift and once during day shift, by the reconciliation team. This bulk insert process will follow the same steps as bulk importing the historical data for testing purposes through Access. Once the vendor database is live, another SER and PDD will be created and submitted to implement a data feed that will be fed directly into the vendor database.

Finally, the logistic clerk will select their name from the drop down and click “save movelog.” This will save the new movelog information onto the SQL server, and the logistic clerk can repeat the process to scan additional movelogs into the vendor database. The SQL Code that runs this process can be found in Image 6 – SQL for Scan a Movelog Form (below).

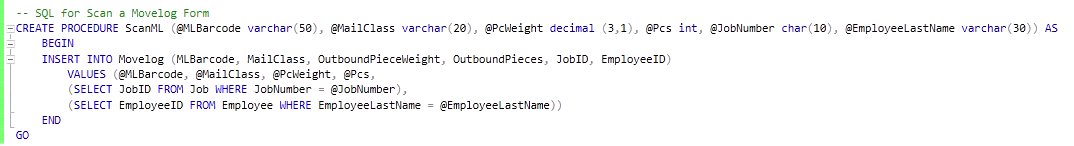


Image 6 – SQL for Scan a Movelog Form

As mentioned, any changes made in Access will be reflected on the SQL Server. Image 6 (below) demonstrates this.

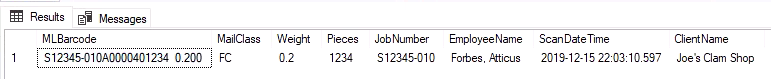


Image 7 – Movelog Form Changed Implemented on SQL Server

1. **Assigning a Movelog to a Consolidated Skid, Load, and Vendor**

Additional forms will be created to assign movelogs to a consolidated skid, and consolidated skids to a load and vendor. The SQL code to implement these actions can be found in Image 8 (below).

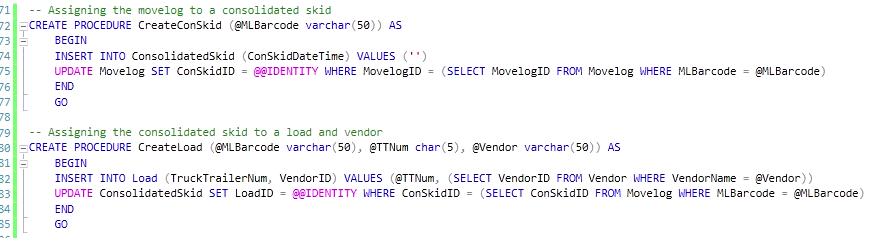


Image 8 – SQL code of completing a movelog in the vendor database

1. **Proof of Mailing for Client Services**

When the reconciliation team receives a request for proof of mailing from client services they can quickly and efficiently look up the job using the proof of mailing report. A proof of mailing requires the vendor, mail class, mail date, and number of pieces mailed. The MLBarcode is also useful, as it represents one pallet of the entire job. For large jobs, Broadridge needs this granular information to ensure all pallets of the job were sent to the correct vendors and by the must mail date.

At Broadridge, a must mail date represents the day the job must mail according to contractual agreements between the client and Broadridge. For internal purposes, when the logistics team needs the “mail date” the load date is used. This is approximately the date and time the truck or trailer left the facility, and therefore represents the Broadridge mailing date.

The rec team will open the report, right click over the job number column, and choose text filters 🡪 contains 🡪 and enter the desired job number. Using the test job S12345, proof of mailing is provided in Image 8 – Proof of Mailing (below).



Image 8 – Proof of Mailing

1. **Outbound Reporting for Invoice Approval**

Before approving invoices, the rec team must reconcile Broadridge records against the invoice. For this, a report needs to be generated that can be filtered on load date (also referred to as mail date), vendor, and mail class. Once filtered, the report will need to return the total number of pieces per mail date and per invoice period. A report called outbound data has been created to do just that. As with the proof of mailing report, the reconciliation team can filter on values to view the report for the desired vendor, time period, and mail class, in order to reconcile against specific invoices. If there is a discrepancy between Broadridge and vendor records, the report also contains job number for a more granular view. In addition, pieces weight is included to verify rates, as vendor rates depend on mail class and piece weight.

Image 9 – Outbound Data (below) displays the outbound data for the test job S12345. Note: it is not the intention to use this report to filter on job number alone, this is being done for demonstration purposes as to display the Mail Date Total and Invoice Period Total.

Once the report is filtered as desired, the reconciliation team will verify that the Invoice Period Total pieces is within five percent variance of the actual invoice received by the vendor. If the variance is greater than five percent, additional research must be completed using the individual job numbers.

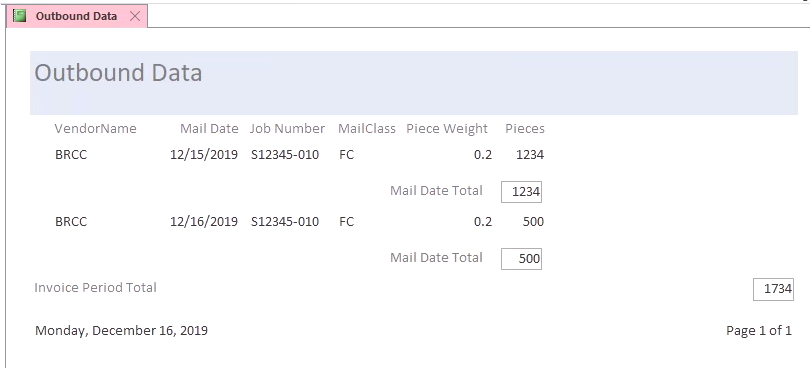


Image 9 – Outbound Data

1. **Average Processing Time**

The director of logistics would like to know, on average, how long it takes the team to process a pallet of mail. As in, how long from the first scan into the vendor database (ScanDateTime) until the mail leaves the facility (LoadDateTime). Image 10 – Logistics’ Average Processing Time outlines the SQL code that will produce this number.

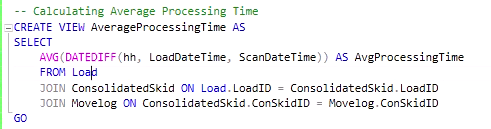


Image 10 – Logistics’ Average Processing Time SQL

According to historical data, the average time for a skid to leave logistics after it has arrived is 41 hours (Image 11 – Logistics’’ Average Processing Time in Access). This is not ideal for an environment with strict mailing dates. The director plans to analyze the process and decrease average processing time and will check this view again in six months.

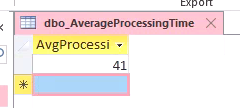


Image 11 – Logistics Average Processing Time in Access

1. **User Permissions**

Before the vendor database goes live, the team will need to be divided into three user groups to allow permissions in a structured manner.

Logistics clerks will receive access to the Scan a Movelog form, as well as additional forms necessary to complete the processing of a pallet.

The reconciliation team will have permission to insert, update, and delete records from the database, but only within strict limits and through a controlled form. The reconciliation team needs these permissions to adjust the data. For example, if records mistakenly did not correct for spoils, there was a mis-scan or bypass (as in, a movelog was never scanned into the database), or if the mail was returned to Broadridge by the vendor and needs to be removed from our records,

The director and other logistic leaders will have access to everything the logistic clerks and reconciliation team has. In addition, certain additional permissions will be granted for reporting and performance purposes, such as the average processing time view.

1. **Summary**

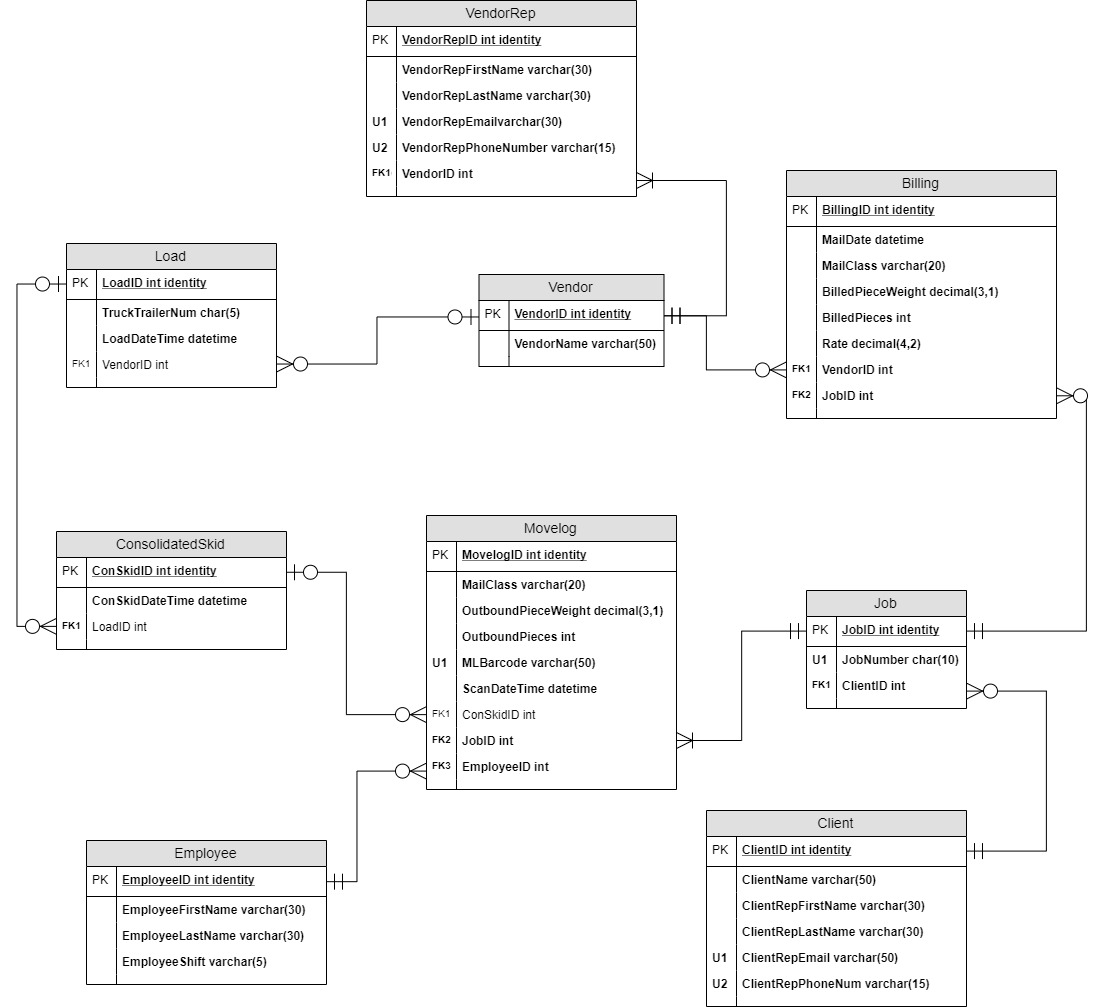
After working with the logistics team and understanding their existing mailing procedures, a functional database has been created to track all skids of mail presented to logistics up until they leave the facility for a presort vendor. Logistic clerks now have a means of electronically inputing move logs using the scan a movelog form, assigning movelogs to a consolidated skid, and assigning consolidated skids to a load for a vendor.

The director can now evaluate the efficiency of the mailing process using the average processing time view, can generate additional reports to trend inefficiencies, and work to improve the process.

The Reconciliation Team can now confirm when and how jobs were mailed at the skid level to provide proof of mailing for Client Services, they can also reconcile outbound piece counts with vendor invoices sent by the vendor representatives.

1. **Appendix**

Appendix 1 – Logical Model



Appendix 2 – SQL Code with result sets

-- Amanda Austin: Final Project for IST659

--Vendor Database for Broadridge Financial Solutions

-- Dropping all tables, if they exist, to avoid errors from duplicate table names

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Movelog')

BEGIN

DROP TABLE Movelog

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Job')

BEGIN

DROP TABLE Job

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Client')

BEGIN

DROP TABLE Client

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Employee')

BEGIN

DROP TABLE Employee

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'ConsolidatedSkid')

BEGIN

DROP TABLE ConsolidatedSkid

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Load')

BEGIN

DROP TABLE Load

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'VendorRep')

BEGIN

DROP TABLE VendorRep

END

GO

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'Vendor')

BEGIN

DROP TABLE Vendor

END

GO

-- Creating the tables from the logical model (excluding billing)

CREATE TABLE Client (

ClientName varchar(50) not null,

ClientRepFirstName varchar(30) not null,

ClientRepLastName varchar(30) not null,

ClientRepEmail varchar(50) not null,

ClientRepPhoneNum varchar(15) not null,

ClientID int identity,

CONSTRAINT PK\_Client PRIMARY KEY (ClientID),

CONSTRAINT U1\_Client UNIQUE (ClientRepEmail),

CONSTRAINT U2\_Client UNIQUE (ClientRepPhoneNum))

CREATE TABLE Job (

JobNumber char(10) not null,

ClientID int not null,

JobID int identity,

CONSTRAINT PK\_Job PRIMARY KEY (JobID),

CONSTRAINT U1\_Job UNIQUE (JobNumber),

CONSTRAINT FK\_Job FOREIGN KEY (ClientID) REFERENCES Client(ClientID))

CREATE TABLE Employee (

EmployeeFirstName varchar(30) not null,

EmployeeLastName varchar(30) not null,

EmployeeShift varchar(5) not null,

EmployeeID int identity,

CONSTRAINT PK\_Employee PRIMARY KEY (EmployeeID))

CREATE TABLE Vendor (

VendorName varchar(50) not null,

VendorID int identity,

CONSTRAINT PK\_Vendor PRIMARY KEY (VendorID))

CREATE TABLE VendorRep (

VendorRepFirstName varchar(30) not null,

VendorRepLastName varchar(30) not null,

VendorRepEmail varchar(30) not null,

VendorRepPhoneNum varchar(15) not null,

VendorID int not null,

VendorRepID int identity,

CONSTRAINT PK\_VendorRep PRIMARY KEY (VendorRepID),

CONSTRAINT U1\_VendorRep UNIQUE (VendorRepEmail),

CONSTRAINT U2\_VendorRep UNIQUE (VendorRepPhoneNum),

CONSTRAINT FK1\_VendorRep FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID))

CREATE TABLE Load (

TruckTrailerNum char(5) not null,

LoadDateTime datetime not null DEFAULT GETDATE(),

VendorID int,

LoadID int identity,

CONSTRAINT PK\_Load PRIMARY KEY (LoadID),

CONSTRAINT FK1\_Load FOREIGN KEY (VendorID) REFERENCES Vendor(VendorID))

CREATE TABLE ConsolidatedSkid (

ConSkidDateTime datetime not null DEFAULT GETDATE(),

LoadID int,

ConSkidID int identity,

CONSTRAINT PK\_ConsolidatedSkid PRIMARY KEY (ConSkidID),

CONSTRAINT FK1\_ConsolidatedSkid FOREIGN KEY (LoadID) REFERENCES Load(LoadID))

CREATE TABLE Movelog (

MailClass varchar(20) not null,

OutboundPieceWeight decimal(3,1) not null,

OutboundPieces int not null,

MLBarcode varchar(50) not null,

ScanDateTime datetime not null DEFAULT GETDATE(),

ConSkidID int,

JobID int not null,

EmployeeID int not null,

MovelogID int identity,

CONSTRAINT PK\_Movelog PRIMARY KEY (MovelogID),

CONSTRAINT U1\_Movelog Unique (MLBarcode),

CONSTRAINT FK1\_Movelog FOREIGN KEY (ConSkidID) REFERENCES ConsolidatedSkid(ConSkidID),

CONSTRAINT FK2\_Movelog Foreign Key (JobID) REFERENCES Job(JobID),

CONSTRAINT FK3\_Movelog FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID))

-- Ensure all data was imported into SQL through Access

SELECT \* FROM Movelog

SELECT \* FROM Job

SELECT \* FROM Client

SELECT \* FROM Employee

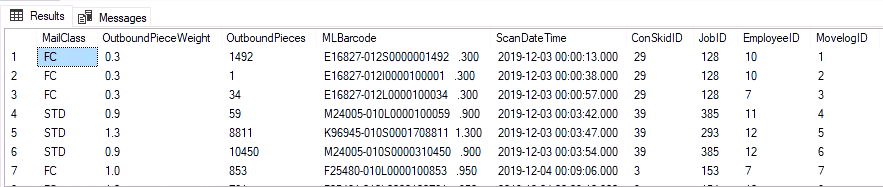
SELECT \* FROM ConsolidatedSkid

SELECT \* FROM Load

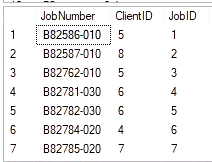
SELECT \* FROM Vendor

SELECT \* FROM VendorRep

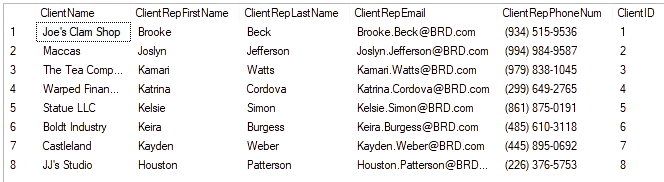
Appendix Image 1 – Movelog Data in SQL (not all data is shown)



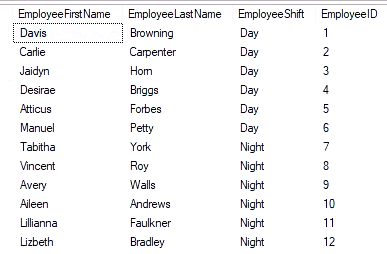
Appendix Image 2 – Job Data in SQL (not all data is shown)



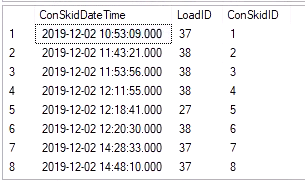
Appendix Image 3 – Client Data in SQL (NOTE: Data has been randomly generated)



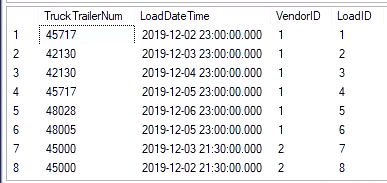
Appendix Image 4 – Employee Data in SQL (NOTE: Data has been randomly generated)



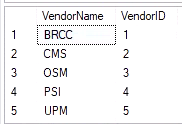
Appendix Image 5 – Consolidated Skid Data in SQL (not all data is shown)



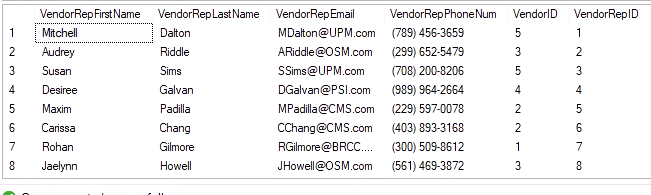
Appendix Image 6 – Load Data in SQL (not all data is shown)



Appendix Image 7 – Vendor Data in SQL



Appendix Image 8 – Vendor Rep Data in SQL (not all data is shown) (NOTE: data has been randomly generated)



-- Inserting new job number for testing purposes

-- Once go live, jobs will be entered into the database through Access

INSERT INTO Job (JobNumber, ClientID) VALUES ('S12345-010', 1)

GO

-- Verifying the movelog for test job S12345 was scanned into the vendor DB from Access

SELECT

MLBarcode,

MailClass,

OutboundPieceWeight AS Weight,

OutboundPieces AS Pieces,

JobNumber,

CONCAT (EmployeeLastName,', ',EmployeeFirstName) AS EmployeeName,

ScanDateTime,

ClientName

FROM Employee

JOIN Movelog ON Employee.EmployeeID = Movelog.EmployeeID

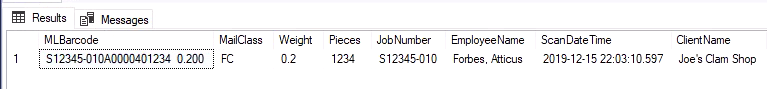
JOIN Job ON Movelog.JobID = Job.JobID

JOIN Client ON Job.CLientID = Client.ClientID

WHERE MLBarcode = 'S12345-010A0000401234 0.200'

GO

Appendix Image 9 – Verify first movelog from test job S12345-010 was inputted into SQL from Access



-- SQL for Scan a Movelog Form

CREATE PROCEDURE ScanML (@MLBarcode varchar(50), @MailClass varchar(20), @PcWeight decimal (3,1), @Pcs int, @JobNumber char(10), @EmployeeLastName varchar(30)) AS

BEGIN

INSERT INTO Movelog (MLBarcode, MailClass, OutboundPieceWeight, OutboundPieces, JobID, EmployeeID)

VALUES (@MLBarcode, @MailClass, @PcWeight, @Pcs,

(SELECT JobID FROM Job WHERE JobNumber = @JobNumber),

(SELECT EmployeeID FROM Employee WHERE EmployeeLastName = @EmployeeLastName))

END

GO

-- Assigning the movelog to a consolidated skid

CREATE PROCEDURE CreateConSkid (@MLBarcode varchar(50)) AS

BEGIN

INSERT INTO ConsolidatedSkid (ConSkidDateTime) VALUES ('')

UPDATE Movelog SET ConSkidID = @@IDENTITY WHERE MovelogID = (SELECT MovelogID FROM Movelog WHERE MLBarcode = @MLBarcode)

END

GO

-- Assigning the consolidated skid to a load and vendor

CREATE PROCEDURE CreateLoad (@MLBarcode varchar(50), @TTNum char(5), @Vendor varchar(50)) AS

BEGIN

INSERT INTO Load (TruckTrailerNum, VendorID) VALUES (@TTNum, (SELECT VendorID FROM Vendor WHERE VendorName = @Vendor))

UPDATE ConsolidatedSkid SET LoadID = @@IDENTITY WHERE ConSkidID = (SELECT ConSkidID FROM Movelog WHERE MLBarcode = @MLBarcode)

END

GO

-- Adding a second movelog from test job S12345 to the database

EXEC ScanML 'S12345-010S0000300500 0.200', 'FC', '0.2', '500', 'S12345-010', 'Forbes'

EXEC CreateConSkid 'S12345-010S0000300500 0.200'

EXEC CreateLoad 'S12345-010S0000300500 0.200', 42005, 'BRCC'

-- Dropping the view, if it exists, to avoid errors from duplicate names

IF EXISTS (SELECT \* FROM INFORMATION\_SCHEMA.TABLES WHERE TABLE\_NAME = 'AverageProcessingTime')

BEGIN

DROP VIEW AverageProcessingTime

END

GO

-- Create view to calculte Average Processing Time

CREATE VIEW AverageProcessingTime AS

SELECT

AVG(DATEDIFF(hh, LoadDateTime, ScanDateTime)) AS AvgProcessingTime

FROM Load

JOIN ConsolidatedSkid ON Load.LoadID = ConsolidatedSkid.LoadID

JOIN Movelog ON ConsolidatedSkid.ConSkidID = Movelog.ConSkidID

GO

-- Return view average processing time

SELECT \* FROM AverageProcessingTime

Appendix Image 10 – Average Processing Time view in SQL

