University of Oklahoma
Big Data
Olivia Long, Evan Langenberg, Karen Li, Marshall Mann, Quannah Lindsey
MIS 3353 – Database Management
Swetha Siripurapu
October 23, 2020

"Don't compromise on excellence."

Executive Summary

The purpose of this project is to provide the Elysian Fly Company with a new relational database. This new database will allow Elysian Fly Company to improve their efficiency, customer segmentation, and ultimately increase profitability. The database built by our company should allow Elysian Fly Company to do their overall business better and therefore give customers a better experience. The scope of this project includes an Entity Relationship Diagram along with assumptions made to form the ERD.

Customer segmentation has been a key issue that needed to be addressed. This project allows for easier customer segmentation by giving customers, orders, and trips attributes that will be put into the database. The ERD created has set the groundwork for building the Elysian Fly Company database. With all of the data collected, Elysian Fly Company will be able to output reports on all aspects of their company, including but not limited to sales, customers, trips, and vendors.

Big Data has collectively logged 16 hours of work into the creation of the ERD. This time logged includes working together as a team as well as individual hours. During this time the team met with the client, made assumptions about the case, completed the ERD design, and prepared a write up. In total phase one of this project is expected to cost Elysian Fly Company $400.

Contents

Get to Know the Team: Big Data

	Name: Evan Langenberg **Major:** Accounting **Minor:** MIS **Year in School:** Senior **Internship Experience:** None yet **Background:** Evan is from Houston, Texas, and has extensive experience with SQL, SAP, Mendix, PowerBI, and VBA.
	Name: Quannah Lindsey **Major:** BBA-MIS, MS-MIT **Year in School:** Senior **Internship Experience:** None **Background:** SQL, PowerBI, Python, VBA, R
	Name: Karen Li **Major:** MIS **Year in School:** Junior **Internship Experience:** None **Background:** Karen is originally from Tulsa, OK. She moved to Norman, OK to pursue a Bachelor's Degree in MIS at the University of Oklahoma. Throughout her college years, Karen has had experience working with SQL, VBA, Excel, and C#.
	Name: Olivia Long **Major:** MIS & Energy Management **Year in School:** Junior **Internship Experience:** None **Background:** From Dallas Area, knowledge in SQL, C#, VBA
	Name: Marshall Mann **Major:** MIS & Energy Management **Year in School:** Senior **Internship Experience:** O&G experience **Background:** from OKC area, SQL, Python, C#

Conceptual Design

The purpose of conceptual design is to visually show the entities, relationships, and attributes of the content of the database that needs to be created. Moreover, through conceptual design, we aim to abstract the structure of our database by analyzing the information we have been given. During this process we will work to construct our rough drafts of an ERD through hashing out assumptions derived from the case. By participating in conceptual design we aim to help Elysian Fly Company become more profitable and efficient by creating a holistic and thorough ERD to meet their needs.

The Client Meeting

This section contains statistics about the client meeting including when we met with the client, where we met with the client, who interviewed the client and the identity of the client interviewed.

- Meeting Time: 10/6/20, 3:00-3:30 pm
- Location: Zoom
- Interviewers: Olivia Long, Quannah Lindsey, Karen Li, Evan Langenberg and Marshall Mann
- Interviewee: Sam Ferreira

Q&A During the Meeting & Information We Learned

The following are the questions we asked during our client meeting, notated as numbers, followed by the answers, indicated by letters.

1. On the guide services section, would you like us to model the different lakes and rivers that the guides will take clients to?
    a. Need to know the location where the trips go.
    b. GPS location calculation is fine
2. Would it be safe to say a customer can only be one of the 3 types?
    a. More than likely yes, locals who are regulars.
    b. Distant customer who moves to the local area can be reclassified
3. What is your criteria for most preferred vendors? Number of sales?
    a. Most preferred vendors would be the number of sales and quality
    b. Once they are in the system they are most likely your vendors until you have to redo their contract
    c. We can have multiple vendors.
    d. A lot of deals and sells a lot of products.
    e. If you are a vendor that does a lot of sales.
    f. You may reorder the preferred vendor before the less preferred vendor because you know the rate of sales.
    g. Reorder points are very important.
4. The effect of flyers on sales
    a. They want email addresses to facilitate marketing of catalogs

b. If it is a repeat customer you can track their purchases. You don’t have to ask the customer for information. If we don’t have customer info they will input it, if they do have information they are a repeat customer.

5. What is the criteria for giving out discounts to customers?
   a. Discounts are mostly seasonal
   b. Sometimes there are seasonal discounts. We come up with discounts when customers go five times, holiday specials, Seasonal visitors are given a discount.
6. Does the firm want to further investigate the pattern between type of customers and type of guides?
   a. A customer is allowed to request a guide
   b. When a customer calls in
   c. Really nice if a customer can request for a guide
   d. If a customer is seasonal, more than likely they are busy during the season and might not get the client they want.
7. Are employees also customers as well
   a. Yes
   b. We tell the employees not to take off during peak season
   c. Employees get a discount
   d. Employees are assigned to multiple roles and are not worried about employees being a customer.
8. Do you record multiple phone numbers and addresses?
   a. One time customer: local address where you are staying
   b. Multiple addresses and phone numbers
9. On the fly bundles, can you do any combo on the bundles or are they limited to one type of fly?
   a. Determine how many flies the customer wants to buy
   b. Normal bundles are plastic fly
   c. You are allowed to customize the combo and say I will take a premade package or customize hooks from a particular vendor and put together the rest of the items in the kit.
   d. DIY bundle where a customer is allowed to put together whatever he wants.
10. What does systematically control purchasing processes mean?
    a. On purchasing,
    b. Purchasing is done by authorized individuals
    c. Needs to make sure there is a person who is doing this
    d. Separation of duties
    e. Who is qualified to make a purchase.
    f. Do not share questions with any of the other teams.
11. What quantitative factors are involved in measuring salespeople and vendor manager success?
    a. Sales measures success for both categories
12. What kind of information would you like us to collect about your customers?
    a. Everything, CC info, etc.
13. DIY bundles is one of the production cycles and you will tell them if your ERD belongs to one of the cycles.

Significant Assumptions

In this section, we will discuss the significant assumptions that helped shape key decisions in designing this ERD. A significant assumption could be quantified as an assumption derived from information in our case study that bears significant weight with regard to particular design choices. The following assumptions will be assigned respective numbers one through five.

1. The **OfferingBundle** is an associative entity that provides a copacetic way to track product bundles such as our lure kit by aggregating several products together and returning a unique product ID to be used in offerings.
2. **CustomerType** is a reference table that was created in order to account for the assumption that customer types are relatively static, as described by the firm. Therefore, when sorting and segmenting our customers, we can sort by the type of customer they are in one singular location.
3. **Fly** is not limited to just flies themselves, the entity Fly can also track other products/parts of kits such as feathers, hooks, flies that can eventually be subcomponents of a Bundle. Fly is set up this way because a Bundle often requires more than just flies.
4. An **Employee** can be either a regular employee, or a guide who leads Customers on Trips. This is denoted by the EmpGuide binary attribute on the Employee table.
5. **SalesChannel** is the reference table that connects to our SalesOrder table and delineates sales based on one of the three sales channels: in-person, phone, or online.
6. We do not track the payment info or **PmtInfo** for our invoices regarding materials purchased. To reflect the payment information of materials we purchased, we simply refer to the cost of those materials which is tracked for each fly as **FlyCost**.

What is an ERD? Why is it necessary?

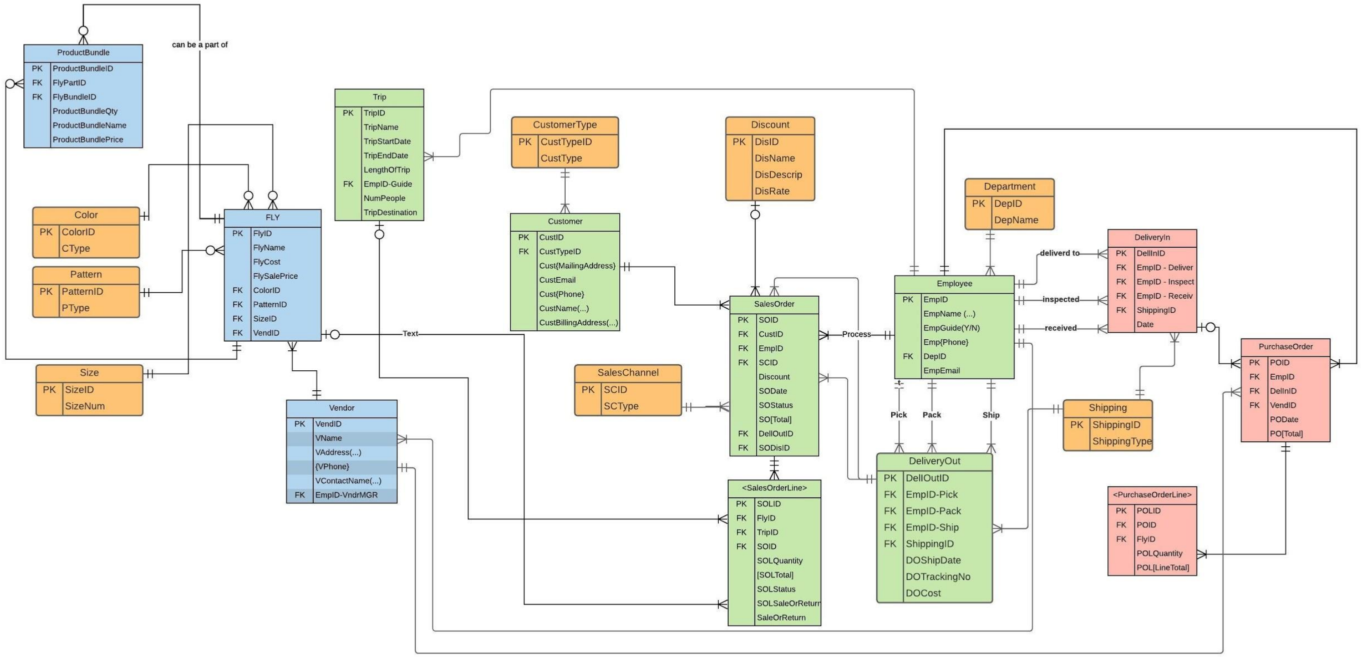
An ERD can be defined as a visual representation of a relational database that shows how each piece fits together. More specifically, it shows how the nouns and verbs of business relate to one another. For example, imagine that you are a manager and want to create a diagram of the team you manage. What is the relationship between the team and the employees that make it up? One team has many employees, but depending on your company's structure, an employee could also be on many teams, or maybe they are only assigned to one at a time. These important distinctions are important components of an ERD, and show why ERDs are so important to showing relationships between data.

Business Cycles Used

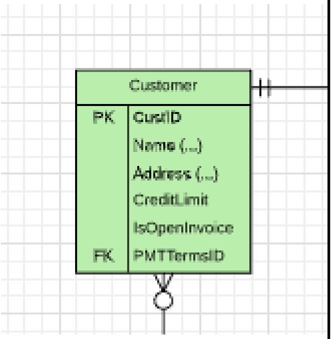
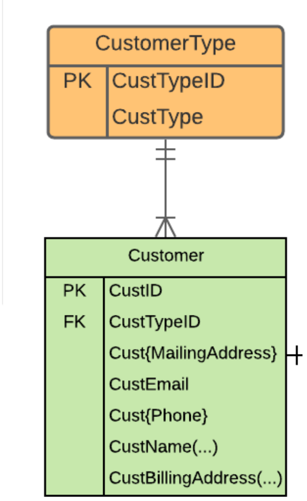
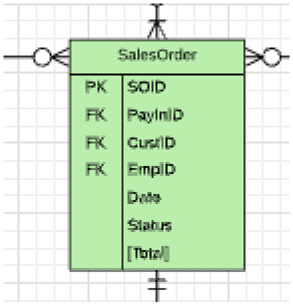
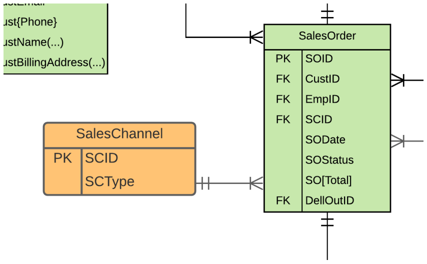
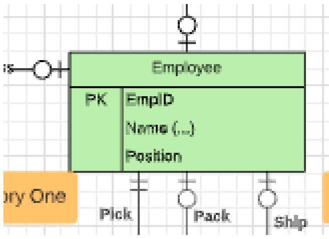
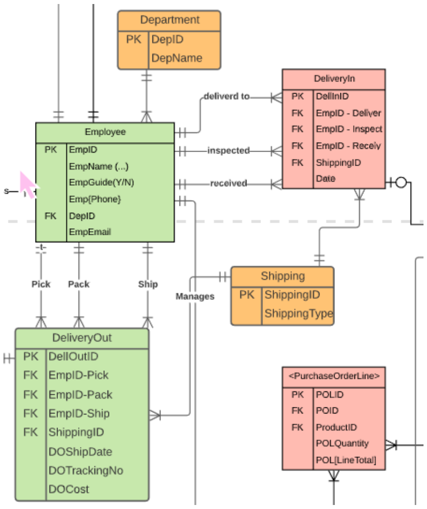
Our ERD incorporated all three business cycles: the revenue cycle, expenditure cycle, and lastly, production cycle. The firm, Elysian, utilizes a modified version of the revenue cycle. The reason that we knew to include this cycle is because Elysian sells both products and services for profit, and thus requires a unique model of their revenue cycle. Elysian also has an expenditure cycle, as the firm does have to procure products and raw materials from vendors to manufacture and sell. Moreover, as previously mentioned, Elysian manufactures goods from raw materials, thus they require a modeling of their production flow in the ERD.

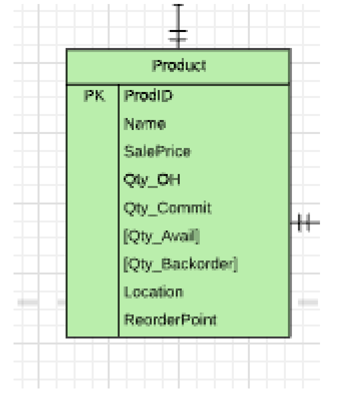
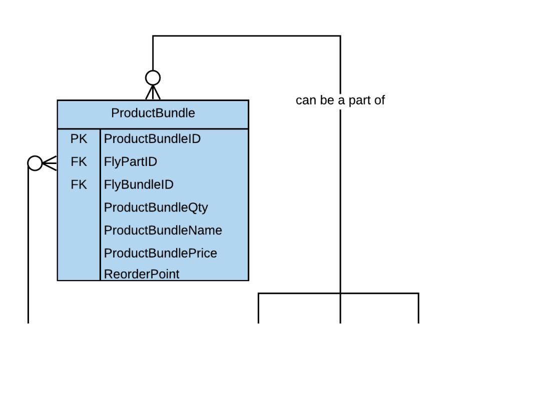
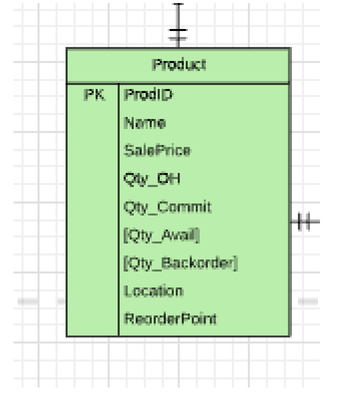
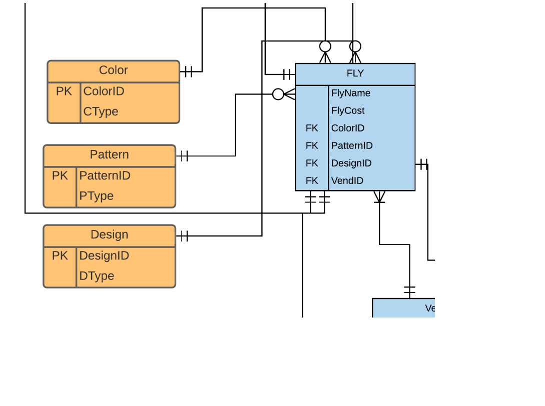
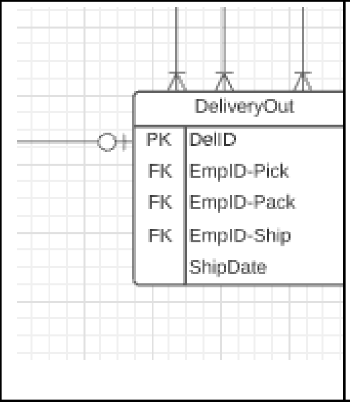
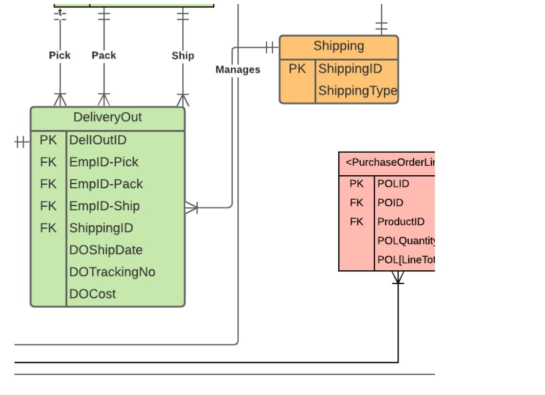
ERD Created

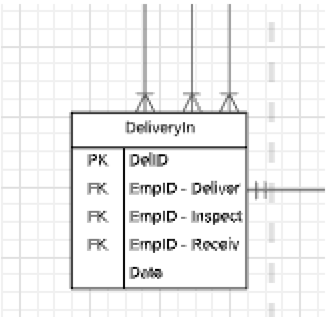
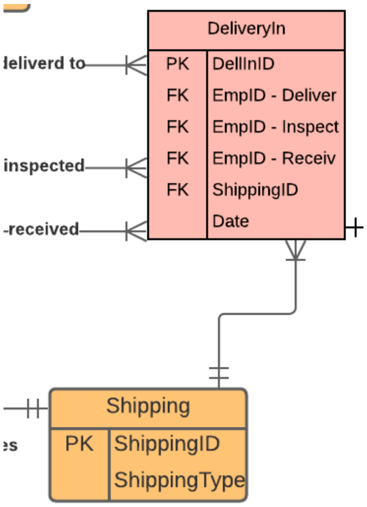
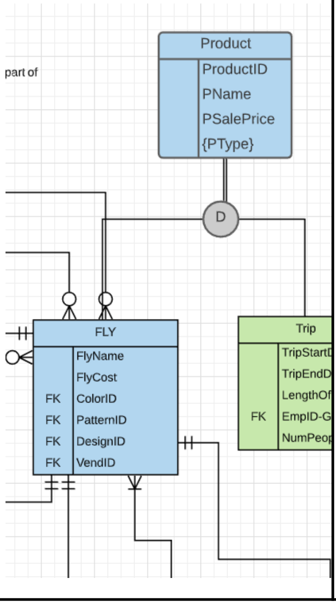
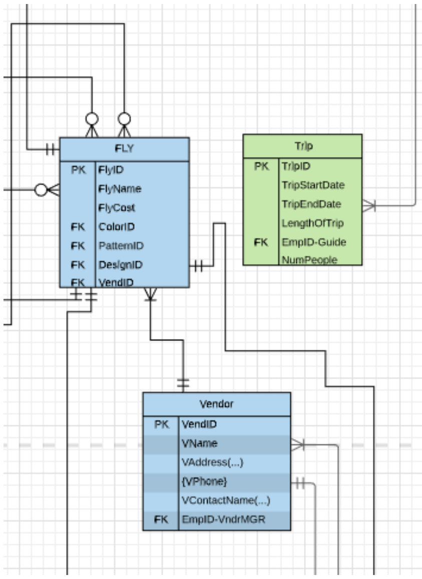
Our ERD:

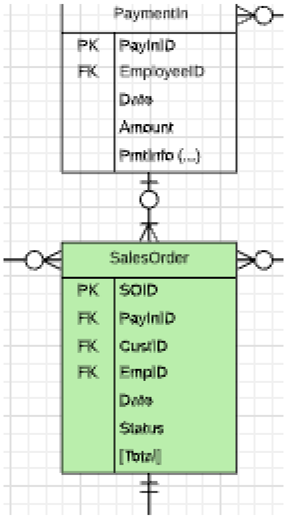
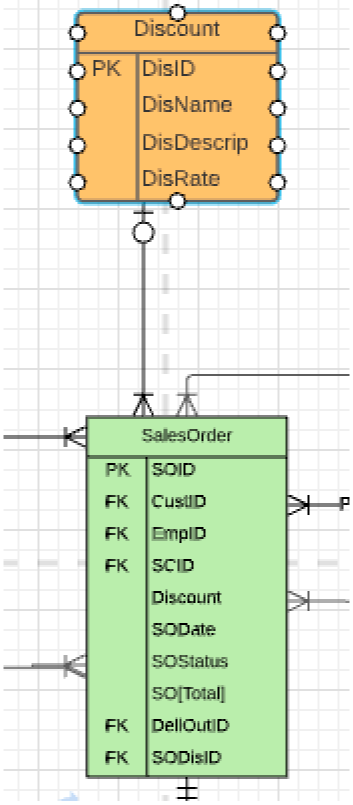


Changes made to generic ERDs

Change #	Original ERD	Updated ERD
Differences b/t Customer Entity: • Customer is now attached to the reference table CustomerType	 Customer PK CustID Name (...) Address (...) CreditLimit IsOpenInvoice FK PMTTermsID	 CustomerType PK CustTypeID CustType Customer PK CustID FK CustTypeID Cust{MailingAddress} CustEmail Cust{Phone} CustName(...) CustBillingAddress(...)
• SalesOrder now has a reference table attached to it named Sales Channel	 SalesOrder PK SOID FK PayInID FK CustID FK EmpID Date Status [Total]	 ıst{Phone} ıstName(...) ıstBillingAddress(...) SalesChannel PK SCID SCType SalesOrder PK SOID FK CustID FK EmpID FK SCID SODate SOStatus SO[Total] FK DelIOutID
• Employee table is connected to the Reference table Department also DeliveryOut, and DeliveryIn.	 Employee PK EmpID Name (...) Position ıry One Pick Pack Ship	 Department PK DepID DepName Employee PK EmpID EmpName (...) EmpGuide(Y/N) Emp{Phone} FK DepID EmpEmail DeliveryIn PK DelInID FK EmpID - Deliver FK EmpID - Inspect FK EmpID - Receiv FK ShippingID Date deliverd to inspected received Pick Pack Ship Manages Shipping PK ShippingID ShippingType DeliveryOut PK DelIOutID FK EmpID-Pick FK EmpID-Pack FK EmpID-Ship FK ShippingID DOShipDate DOTrackingNo DOCost <PurchaseOrderLine> PK POLID FK POID FK ProductID POLQuantity POL[LineTotal]

• Product bundle is an associative entity that bundles products together.	 Product PK ProdID Name SalePrice Qty_OH Qty_Commit [Qty_Avail] [Qty_Backorder] Location ReorderPoint	 ProductBundle PK ProductBundleID FK FlyPartID FK FlyBundleID ProductBundleQty ProductBundleName ProductBundlePrice ReorderPoint can be a part of
• Product now has the color, pattern, and size reference tables attached	 Product PK ProdID Name SalePrice Qty_OH Qty_Commit [Qty_Avail] [Qty_Backorder] Location ReorderPoint	 Color PK ColorID CType Pattern PK PatternID PType Design PK DesignID DType FLY FlyName FlyCost FK ColorID FK PatternID FK DesignID FK VendID Ve
• DeliveryOut is now connected to Shipping.	 DeliveryOut PK DelID FK EmpID-Pick FK EmpID-Pack FK EmpID-Ship ShipDate	 Pick Pack Ship Manages Shipping PK ShippingID ShippingType DeliveryOut PK DelIOutID FK EmpID-Pick FK EmpID-Pack FK EmpID-Ship FK ShippingID DOShipDate DOTrackingNo DOCost <PurchaseOrderLi PK POLID FK POID FK ProductID POLQuantity POL[LineTot

- DeliveryIn is now attached to Shipping.	 DeliveryIn PK DelID FK EmpID - Deliver FK EmpID - Inspect FK EmpID - Receiv Date	 DeliveryIn PK DelInID FK EmpID - Deliver FK EmpID - Inspect FK EmpID - Receiv FK ShippingID Date deliverd to inspected -received Shipping PK ShippingID ShippingType es
- Fly and Trip are now not part of a supertype formerly called Product, and are their own entities.	 Product ProductID PName PSalePrice {PType} part of D FLY FlyName FlyCost FK ColorID FK PatternID FK DesignID FK VendID Trip TripStartD TripEndD LengthOf FK EmpID-G NumPeop	 FLY PK FlyID FlyName FlyCost FK ColorID FK PatternID FK DesignID FK VendID Trip PK TripID TripStartDate TripEndDate LengthOfTrip FK EmpID-Guide NumPeople Vendor PK VendID VName VAddress(...) {VPhone} VContactName(...) FK EmpID-VndrMGR

- SalesOrder has a reference table attached to it named Discount	 PaymentIn PK PayInID FK EmployeeID Date Amount PmtInfo (...) SalesOrder PK SOID FK PayInID FK CustID FK EmpID Date Status [Total]	 Discount PK DisID DisName DisDescrip DisRate SalesOrder PK SOID FK CustID FK EmpID FK SCID Discount SODate SOStatus SO[Total] FK DellOutID FK SODisID

Logical Design

Logical design is the phase where normalization occurs. This phase of logical design serves as a blueprint to guide developers. Through normalization, the logical design phase helps clear up any redundancies and makes the database more efficient for whenever reports are run. In this project, logical design is used by normalizing the relationships created for Elysian Fly Company. For this project form 3NF is used to normalize the data, this means that all columns are atomic, there are no partial functional dependencies, and there are no transitive dependencies. By using 3NF the relations should have no data duplication problems and they should be atomic.

Normalization

Normalization is the process used to ensure that a database outputs reliable reports by attaining data integrity. It also helps ensure that when the database is outputting reports, that it is efficient in that process. Normalization will ensure the atomicity of columns and eliminate data duplication issues.

Normalized Relations (Primary Keys are underlined; Foreign Keys *italicized*)

TColor(ColorID, CType)

TPattern(PatternID, PType)

TSize(SizeID, SizeNum)

TSalesChannel(SCID SCType)

TCustomerType(CustTypeID, CustType)

TDepartment(DepartmentID, DepName)

TShipping(SID, ShippingType)

TEmployee(EmpID, EmpFirstName, EmpLastName, EmpGuide(Y/N), EmpHomePhone, EmpCellPhone, EmpEmail, *EmpDepID*)

Foreign Key EmpDepID references TDepartment
Not Null
On Delete No Action

TVendor(VendorID, VName, VAddress, VPhone, VContactName, *EmpID-VndrMGR)*

Foreign Key references TEmployee

Not Null

On Delete No Action

TFly(FlyID, FlyName, FlySalePriceFlyCost, *ColorID, PatternID, SizeID, VendorID*)

Foreign Key ColorID references TColor

Not Null

On Delete No Action

Foreign Key PatternID references TPattern

Not Null

On Delete No Action

Foreign Key SizeID references TSize

Not Null

On Delete No Action

Foreign Key VendorID references TVendor

Not Null

On Delete No Action

TProductBundle(ProductBundleID, *FlyBundleID, FlyPartID*, ProductBundleQty, ProductBundleName, ProductBundlePrice)

Foreign Key FlyID-Bundle references TFly

Not Null

On Delete No Action

Foreign Key FlyID-Part references TFly

Not Null

On Delete No Action

TTrip(TripID, TripName, TripStartDate, TripEndDate, *EmpID-Guide*, NumPeople)

Foreign Key EmpID-Guide references TEmployee

Not Null

On Delete No Action

TCustomer(CustID, *CustTypeID*, CustMailSteet, CustMailCity, CustMailState, CustMailZip CustEmail, CustCellPhone, CustHomePhone, CustFirstName, CustLastName, CustBillStreet, CustBillCity, CustBillState, CustBillZip)

Foreign Key CustTypeID references TCustomerType
Not Null
On Delete No Action

TDiscount(DisID, DisName, DisDescrip, DisRate)

TSalesOrder(SOID, *SOCustID*, *SOEmpID*, *SOSCID*, *SODelOutID*, *SODisID*, SODate, SOStatus)

Foreign Key SOCustID references TCustomer
Not Null
On Delete No Action
Foreign Key SOEmpID references TEmployee
Not Null
On Delete No Action
Foreign Key SOSCID references TSalesChannel
Not Null
On Delete No Action
Foreign Key SODelOutID references TDeliveryOut
Not Null
On Delete No Action
Foreign Key SODisID references TDiscount
Not Null
On Delete No Action

TSalesOrderLine(SOLID, *SOFlyID*, *SOID*, SOLQuantity, SOLStatus, SOLSaleOrReturn)

Foreign Key SOLID references TSalesOrder
Not Null
On Delete No Action

TDeliveryIn(DelInID, *DelEmpID-Deliver*, *DelEmpID-Inspect*, *DelEmpID-Receive*, *DelInShippingID*, Date)

Foreign Key DIEmpID-Deliver references TEmployee
Not Null
On Delete No Action
Foreign Key DIEmpID-Inspect references TEmployee
Not Null
On Delete No Action
Foreign Key DIEmpID-Receive references TEmployee
Not Null
On Delete (No Action)
Foreign Key DIShippingID references TShipping
Not Null
On Delete No Action

TPurchaseOrder(POID, *POEmpID*, *PODelInID*, *POVendID*, PODate)

Foreign Key PODelInID references TDeliveryIn
Null Allowed
On Delete Set Null
Foreign Key POEmpID references TEmployee
Not Null
On Delete No Action
Foreign Key POVendID references TVendor
Not Null
On Delete No Action

TPurchaseOrderLine(POLID, *POPOID,* POLQuantity)

Foreign Key POPOID references TPurchaseOrder
Not Null
On Delete No Action

TDeliveryOut(DelOutID, *DelEmpID-Pick*, *DelEmpID-Pack*, *DelEmpID-Ship*, *DelOutShippingID*, Date)

Foreign Key DOEmpID-Pick references TEmployee
Not Null
On Delete No Action
Foreign Key DoEmpID-Pack references TEmployee
Not Null
On Delete No Action
Foreign Key DoEmpID-Ship references TEmployee
Not Null
On Delete No Action
Foreign Key DOShippingID references TShipping
Not Null
On Delete No Action

Differences between ERD and Normalized Relations

The major difference between an entity relationship diagram and normalized relations is that when normalizing, attributes are assigned to specific fields and relationships are further broken down. For example, a multi value attribute is acceptable to be included in an ERD, but when normalization occurs that multi value attribute has to be taken into parts of the subjects that it represents. Another major difference is that ERDs are a diagram showing the relationships between entities, while normalized relations dig into those relationships and help ensure that there are no redundancies. It is beneficial to have normalized relations to ensure that there will be no redundancy in the data and ensure the atomicity of the data. Ensuring these two factors helps confirm that when reports are outputted, they will be accurate and efficient.

Referential Integrity

Referential Integrity refers to the association between entities. This means that if there is a relationship among entities, the foreign key shown has to match up with a valid primary key, or else it must be null. This ensures that a reference from a row in one table to a row in another table is valid. The formula for Referential Integrity is:

Foreign key [name] references [table name mandatory side]

[Not Null/Null Allowed]

On Delete (Restrict/Cascade/Set Null)

On Update Cascade

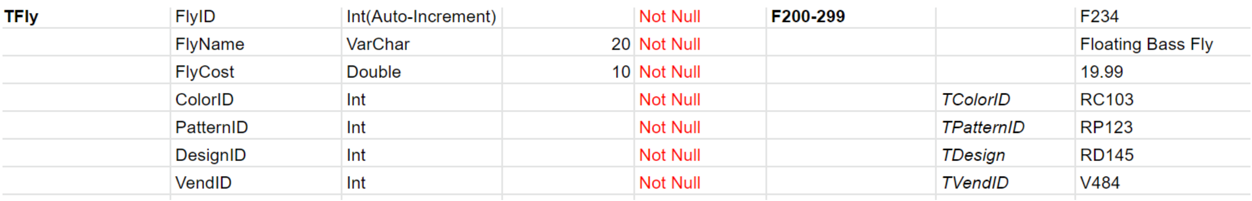
By adding this into the normalization process, the principal end of the constraint, the dependent end of the constraint, and the referencing property on the dependent end is specified.

Physical Design and Implementation

Physical design is a step in the database design process where the design of the database is integrated into a RDBMS. An RDBMS is a relational database management system. The purpose of the physical design aspect helps ensure that the database outputs reports efficiently. Physical design is platform specific which means that the design will vary depending on which server it is built for. This is important for Big Data to use to ensure that Elysian Fly Company has a database that outputs reports correctly and in a timely fashion.

Data Dictionary

A data dictionary is an output resulting from the physical design process. The data dictionary reflects the field name, data type, size (if applicable), whether it is null or not null, the primary key range, references, and sample data. This helps the database developers create sample data, determine how the database should be implemented, and determine how the data fields should be used. Big Data used a data dictionary to help create sample data tables. These tables were then transferred to be used to create the required queries. An example of the data dictionary Big Data used is shown below.



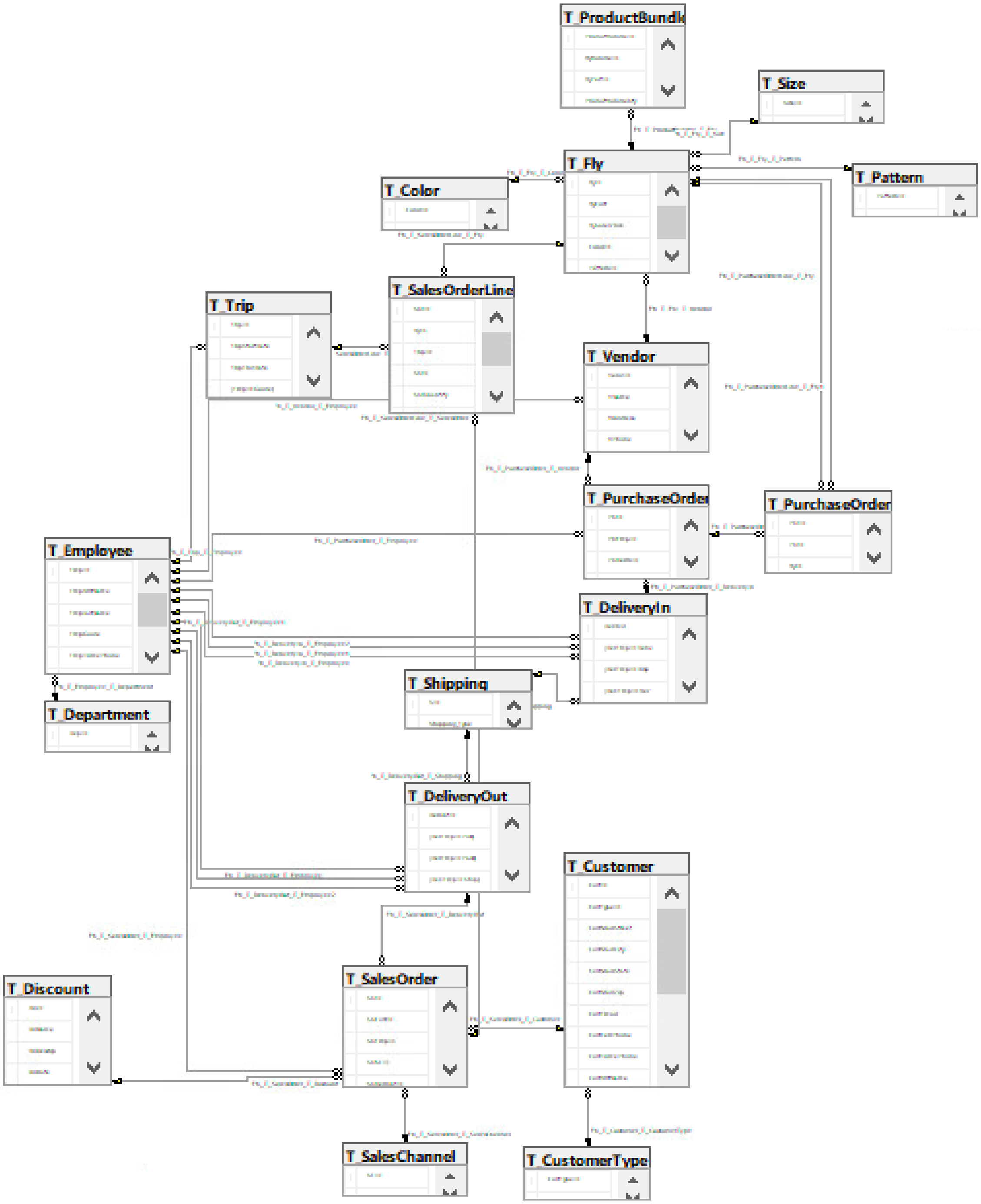
TFly	FlyID	Int(Auto-Increment)		Not Null	**F200-299**		F234
	FlyName	VarChar	20	Not Null			Floating Bass Fly
	FlyCost	Double	10	Not Null			19.99
	ColorID	Int		Not Null		*TColorID*	RC103
	PatternID	Int		Not Null		*TPatternID*	RP123
	DesignID	Int		Not Null		*TDesign*	RD145
	VendID	Int		Not Null		*TVendID*	V484

In this example the table TFly is shown. The attributes of Fly are listed along with each of their data types. For FlyName and FlyCost, the sizes are shown because they are a varchar and double. The “F200-299” represents the primary key range. Color, Pattern, Design, and Vend ID show where they are referenced from since they are foreign keys. The far-right column shows examples of how the data will be displayed. All of these components are helpful to give developers a visual of the data and make the overall database creation process easier.

Denormalization

Denormalization is the process of reducing the normal form of tables within the database. The process of denormalization allows data duplication, which is the reverse of what normalization does. Denormalization allows for the database to process queries more efficiently. Because denormalization allows for data duplication and condensing fields, problems may arise when using this process. There are guidelines as to whether to implement denormalization or not. These guidelines are if the entries for a denormalized field are the same, and if a reference table is referenced by only one other table and that its attributes are not usually included in a WHERE clause.

Implemented Physical Design



Challenges Faced/Addressed During Implementation

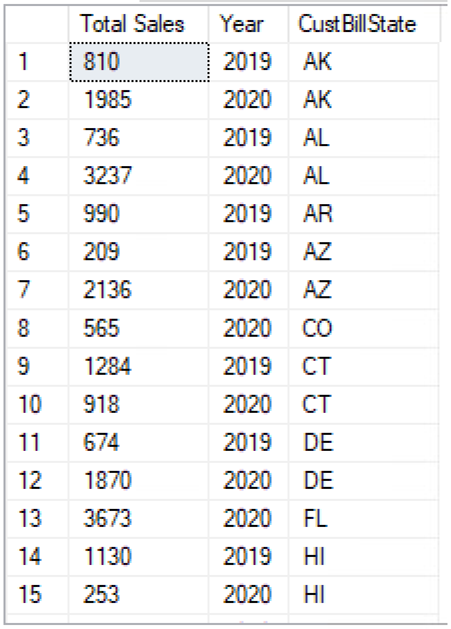
Challenges that we faced included missing attributes within our data once we started writing queries, deciding whether or not to implement denormalization, and making sure everything was cohesive. We overcame the missing attributes by adding attributes into our database to ensure our queries could run effectively. When determining whether to normalize or denormalize we also evaluated our data to see if we needed to have it in 2NF or 3NF to make our database more accessible.

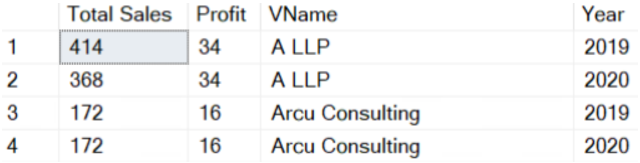
Strengths and Weaknesses Encountered During Implementation

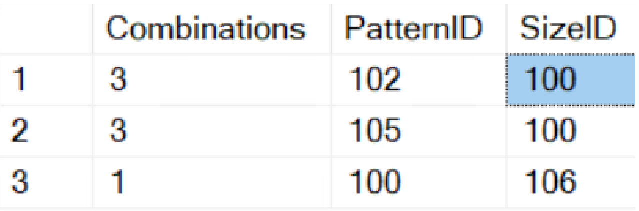
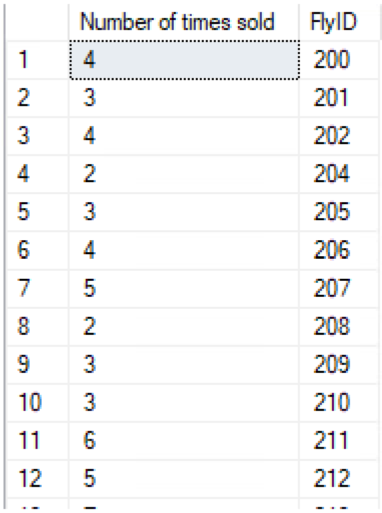
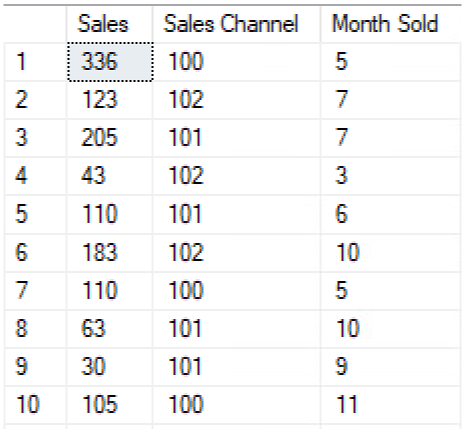
Strengths that Big Data possessed throughout implementation included creating the data dictionary, working well together, and testing our queries efficiently. Some weaknesses of Big Data while building the ERDs and writing queries when implementing this database included creating the original queries, recalling information from previous modules, and working with a new database system that was unfamiliar to us.

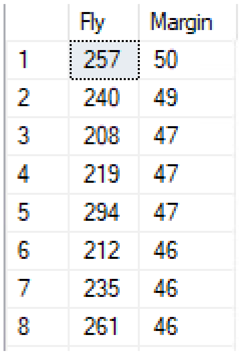
Specific SQL Statements Requested

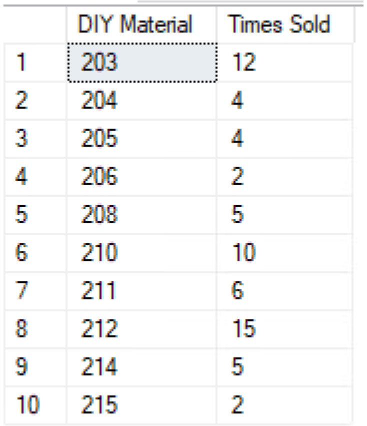
In this section we have displayed the specific programs we were asked to execute by Elysian Fly Company. These queries were written and tested in Microsoft SQL Server Management Studio by the team. Included below is the question asked, the SQL code, and a glimpse of the output is displayed to give a visual on what the report ran.

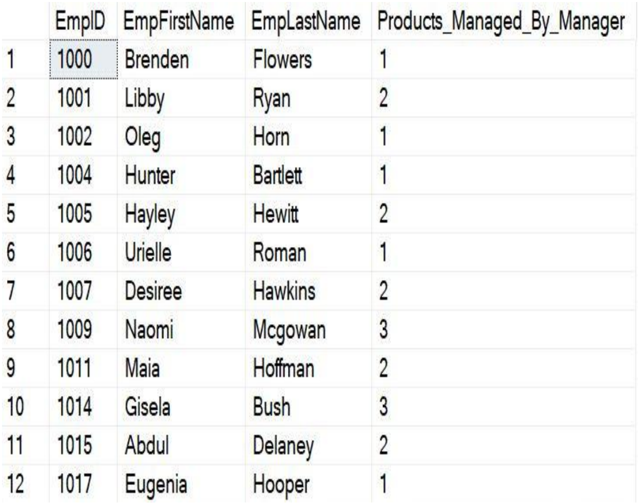
Query #	Question	SQL	Partial Output
1.	What is the total sales(in dollars) by customer state per year?	SELECT SUM((FlySalesPrice*SOLQuantity)+(T.Price*SOLQuantity)) as "Total Sales", Year(SODate) as "Year", CustBillState FROM T_Customer C JOIN T_SalesOrder SO ON C.CustID=SO.SOCustID JOIN T_SalesOrderLine SOL ON SO.SOID=SOL.SOID JOIN T_Fly F ON SOL.FlyID=F.FlyID JOIN T_Trip T ON SOL.TripID = T.TripID GROUP BY Year(SODate), CustBillState	 Total Sales · Year · CustBillState 1 · 810 · 2019 · AK 2 · 1985 · 2020 · AK 3 · 736 · 2019 · AL 4 · 3237 · 2020 · AL 5 · 990 · 2019 · AR 6 · 209 · 2019 · AZ 7 · 2136 · 2020 · AZ 8 · 565 · 2020 · CO 9 · 1284 · 2019 · CT 10 · 918 · 2020 · CT 11 · 674 · 2019 · DE 12 · 1870 · 2020 · DE 13 · 3673 · 2020 · FL 14 · 1130 · 2019 · HI 15 · 253 · 2020 · HI

2.	What is the total sales(in dollars) by vendor per year? We must be able to calculate profit.	SELECT
		SUM(SOLQuantity*FlySalesPrice) as "Total Sales", (FlySalesPrice-FlyCost) as Profit, VName, Year(SODate) as "Year"
		FROM T_SalesOrder SO
		JOIN T_SalesOrderLine SOL ON SO.SOID=SOL.SOID Join T_Fly F ON SOL.FlyID=F.FlyID JOIN T_Vendor V ON F.VendID=V.VendID
		GROUP BY VName, FlySalesPrice, FlyCost, Year(SODate)
			
			Total Sales · Profit · VName · Year
			1 · 414 · 34 · A LLP · 2019
			2 · 368 · 34 · A LLP · 2020
			3 · 172 · 16 · Arcu Consulting · 2019
			4 · 172 · 16 · Arcu Consulting · 2020

<table>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td>3.</td><td>What are the ten highest selling patterns, sizes, and pattern-size-color combinations in a given year?</td><td>SELECT Top 10 Count(FlyBundleID) as "Combinations", PatternID, SizeID, ColorID FROM T_SalesOrder SO Join T_SalesOrderLine SOL ON SO.SOID=SOL.SOID JOIN T_Fly F ON SOL.FlyID=F.FlyID JOIN T_ProductBundle PB ON F.FlyID=PB.FlyPartID WHERE Year(SODate)=2019 GROUP BY PatternID, SizeID ColorID</td><td>
<table>
<tr><th></th><th>Combinations</th><th>PatternID</th><th>SizeID</th></tr>
<tr><td>1</td><td>3</td><td>102</td><td>100</td></tr>
<tr><td>2</td><td>3</td><td>105</td><td>100</td></tr>
<tr><td>3</td><td>1</td><td>100</td><td>106</td></tr>
</table></td></tr>
<tr><td>4.</td><td>What is the number of times each product(fly) was sold? We want to see also those flies that have never been sold so that we can discontinue them.</td><td>SELECT F.FlyID, Count(F.FlyID) FROM T_Fly F, T_SalesOrderLine SOL WHERE F.FlyID=SOL.FlyID AND (SELECT F.FlyID FROM T_Fly AS Fly, T_SalesOrderLine AS SOL WHERE SOL.FlyID = Fly.FlyID AND SOL.FlyID = NULL)</td><td>
<table>
<tr><th></th><th>Number of times sold</th><th>FlyID</th></tr>
<tr><td>1</td><td>4</td><td>200</td></tr>
<tr><td>2</td><td>3</td><td>201</td></tr>
<tr><td>3</td><td>4</td><td>202</td></tr>
<tr><td>4</td><td>2</td><td>204</td></tr>
<tr><td>5</td><td>3</td><td>205</td></tr>
<tr><td>6</td><td>4</td><td>206</td></tr>
<tr><td>7</td><td>5</td><td>207</td></tr>
<tr><td>8</td><td>2</td><td>208</td></tr>
<tr><td>9</td><td>3</td><td>209</td></tr>
<tr><td>10</td><td>3</td><td>210</td></tr>
<tr><td>11</td><td>6</td><td>211</td></tr>
<tr><td>12</td><td>5</td><td>212</td></tr>
</table></td></tr>
<tr><td>5.</td><td>Total sales (in dollars) for each channel per month</td><td>SELECT (SOL.SOLQuantity*Fly.FlySalesPrice) as Sales, SO.SOSCID as [Sales Channel], Month(SO.SODate) as Month FROM T_SalesOrderLine as SOL, T_Fly as Fly, T_SalesOrder as SO WHERE Fly.FlyID = SOL.FlyID AND SOL.SOID = SO.SOID GROUP BY (SOL.SOLQuantity*Fly.FlySalesPrice), SO.SOSCID, Month(SO.SOdate);</td><td>
<table>
<tr><th></th><th>Sales</th><th>Sales Channel</th><th>Month Sold</th></tr>
<tr><td>1</td><td>336</td><td>100</td><td>5</td></tr>
<tr><td>2</td><td>123</td><td>102</td><td>7</td></tr>
<tr><td>3</td><td>205</td><td>101</td><td>7</td></tr>
<tr><td>4</td><td>43</td><td>102</td><td>3</td></tr>
<tr><td>5</td><td>110</td><td>101</td><td>6</td></tr>
<tr><td>6</td><td>183</td><td>102</td><td>10</td></tr>
<tr><td>7</td><td>110</td><td>100</td><td>5</td></tr>
<tr><td>8</td><td>63</td><td>101</td><td>10</td></tr>
<tr><td>9</td><td>30</td><td>101</td><td>9</td></tr>
<tr><td>10</td><td>105</td><td>100</td><td>11</td></tr>
</table></td></tr>
</table>

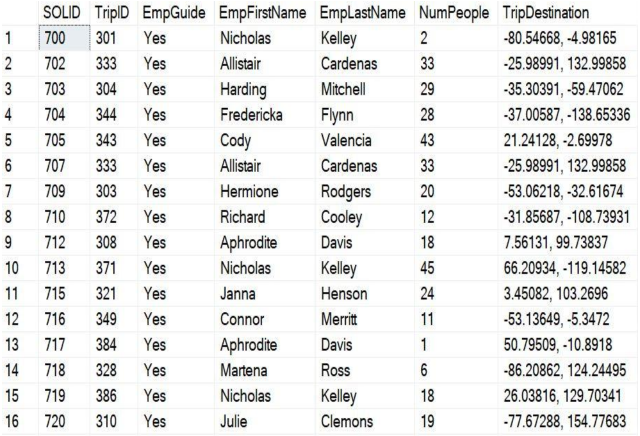
6.	10% of the products that have the highest margin.	SELECT TOP 10 PERCENT WITH TIES Fly.FlyID AS ProductID, (Fly.FlySalesPrice-Fly.FlyCost) as Margin
		FROM T_Fly as Fly
		ORDER BY [Margin] desc;
			
			Fly Margin
			1 257 50
			2 240 49
			3 208 47
			4 219 47
			5 294 47
			6 212 46
			7 235 46
			8 261 46

7.	What are the ten most popular (units sold) DIY flying materials?	SELECT TOP 10 Fly.FlyID as [DIY Material], COUNT(SOL.FlyID) as [Times Sold]
		FROM T_Fly as Fly, T_ProductBundle as PB, T_SalesOrderLine as SOL
		WHERE Fly.FlyID = PB.FlyPartID
		AND PB.FlyPartID = SOL.FlyID
		GROUP BY Fly.FlyID;
			
			DIY Material Times Sold
			1 203 12
			2 204 4
			3 205 4
			4 206 2
			5 208 5
			6 210 10
			7 211 6
			8 212 15
			9 214 5
			10 215 2

8	The number of distinct products managed by each vendor manager.	SELECT E.EmpID, E.EmpFirstName, E.EmpLastName, COUNT(F.FlyID) Products_Managed_By_Manager
		
		FROM T_Employee E JOIN T_Vendor V ON E.EmpID=V[EmpID-VndrMGR] JOIN T_Fly F ON V.VendID=F.VendID
		
		GROUP BY E.EmpID, E.EmpFirstName, E.EmpLastName
			
			EmpID EmpFirstName EmpLastName Products_Managed_By_Manager
			1 1000 Brenden Flowers 1
			2 1001 Libby Ryan 2
			3 1002 Oleg Horn 1
			4 1004 Hunter Bartlett 1
			5 1005 Hayley Hewitt 2
			6 1006 Urielle Roman 1
			7 1007 Desiree Hawkins 2
			8 1009 Naomi Mcgowan 3
			9 1011 Maia Hoffman 2
			10 1014 Gisela Bush 3
			11 1015 Abdul Delaney 2
			12 1017 Eugenia Hooper 1

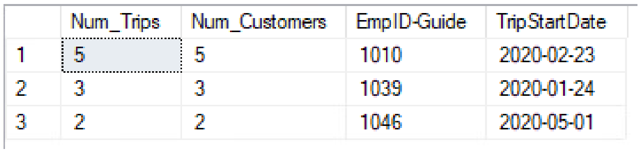
<table>
<tr>
<td>9</td>
<td>The upcoming, scheduled guided trips (i.e., the guided trips that have already been sold) for each guide, including the guide’s name, the trip destination, the customer name, and the number in the customer’s party.</td>
<td>SELECT SO.SOLID, T.TripID, E.EmpGuide, E.EmpFirstName, E.EmpLastName, T.NumPeople, T.TripDestination

FROM T_Employee E JOIN T_Trip T ON E.EmpID=T.[EmpID-Guide] JOIN T_SalesOrderLine SO ON T.TripID=SO.TripID

WHERE EmpGuide=’YES’</td>
<td>
<table>
<tr><th></th><th>SOLID</th><th>TripID</th><th>EmpGuide</th><th>EmpFirstName</th><th>EmpLastName</th><th>NumPeople</th><th>TripDestination</th></tr>
<tr><td>1</td><td>700</td><td>301</td><td>Yes</td><td>Nicholas</td><td>Kelley</td><td>2</td><td>-80.54668, -4.98165</td></tr>
<tr><td>2</td><td>702</td><td>333</td><td>Yes</td><td>Allistair</td><td>Cardenas</td><td>33</td><td>-25.98991, 132.99858</td></tr>
<tr><td>3</td><td>703</td><td>304</td><td>Yes</td><td>Harding</td><td>Mitchell</td><td>29</td><td>-35.30391, -59.47062</td></tr>
<tr><td>4</td><td>704</td><td>344</td><td>Yes</td><td>Fredericka</td><td>Flynn</td><td>28</td><td>-37.00587, -138.65336</td></tr>
<tr><td>5</td><td>705</td><td>343</td><td>Yes</td><td>Cody</td><td>Valencia</td><td>43</td><td>21.24128, -2.69978</td></tr>
<tr><td>6</td><td>707</td><td>333</td><td>Yes</td><td>Allistair</td><td>Cardenas</td><td>33</td><td>-25.98991, 132.99858</td></tr>
<tr><td>7</td><td>709</td><td>303</td><td>Yes</td><td>Hermione</td><td>Rodgers</td><td>20</td><td>-53.06218, -32.61674</td></tr>
<tr><td>8</td><td>710</td><td>372</td><td>Yes</td><td>Richard</td><td>Cooley</td><td>12</td><td>-31.85687, -108.73931</td></tr>
<tr><td>9</td><td>712</td><td>308</td><td>Yes</td><td>Aphrodite</td><td>Davis</td><td>18</td><td>7.56131, 99.73837</td></tr>
<tr><td>10</td><td>713</td><td>371</td><td>Yes</td><td>Nicholas</td><td>Kelley</td><td>45</td><td>66.20934, -119.14582</td></tr>
<tr><td>11</td><td>715</td><td>321</td><td>Yes</td><td>Janna</td><td>Henson</td><td>24</td><td>3.45082, 103.2696</td></tr>
<tr><td>12</td><td>716</td><td>349</td><td>Yes</td><td>Connor</td><td>Merritt</td><td>11</td><td>-53.13649, -5.3472</td></tr>
<tr><td>13</td><td>717</td><td>384</td><td>Yes</td><td>Aphrodite</td><td>Davis</td><td>1</td><td>50.79509, -10.8918</td></tr>
<tr><td>14</td><td>718</td><td>328</td><td>Yes</td><td>Martena</td><td>Ross</td><td>6</td><td>-86.20862, 124.24495</td></tr>
<tr><td>15</td><td>719</td><td>386</td><td>Yes</td><td>Nicholas</td><td>Kelley</td><td>18</td><td>26.03816, 129.70341</td></tr>
<tr><td>16</td><td>720</td><td>310</td><td>Yes</td><td>Julie</td><td>Clemons</td><td>19</td><td>-77.67288, 154.77683</td></tr>
</table></td>
</tr>
<tr>
<td>10</td>
<td>Number of trips and the number of customers taken on fishing trips by each guide in the past 6 months.</td>
<td>SELECT COUNT(SOL.TripID) Num_Trips, COUNT(C.CustID) Num_Customers, T.[EmpID-Guide], T.TripStartDate

FROM T_Customer C JOIN T_SalesOrder SO ON C.CustID=SO.SOCustID JOIN T_SalesOrderLine SOL ON SO.SOID=SOL.SOID JOIN T_Trip T ON SOL.TripID=T.TripID

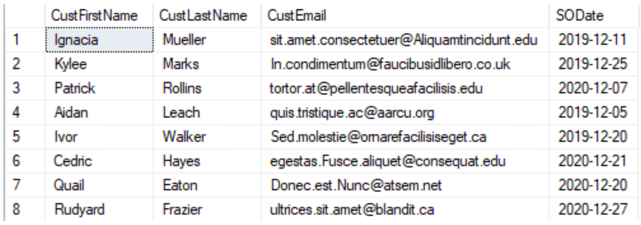
WHERE TripStartDate between ‘2020-01-01’ and ‘2020-06-01’

GROUP BY T.[EmpID-Guide], T.TripStartDate</td>
<td>
<table>
<tr><th></th><th>Num_Trips</th><th>Num_Customers</th><th>EmpID-Guide</th><th>TripStartDate</th></tr>
<tr><td>1</td><td>5</td><td>5</td><td>1010</td><td>2020-02-23</td></tr>
<tr><td>2</td><td>3</td><td>3</td><td>1039</td><td>2020-01-24</td></tr>
<tr><td>3</td><td>2</td><td>2</td><td>1046</td><td>2020-05-01</td></tr>
</table></td>
</tr>
</table>

11

Names and email addresses of all customers who made purchases in a given month. We need to be able to enter the month.

```
CREATE Procedure Query11

@month int

As

SELECT C.CustFirstName, C.CustLastName, C.CustEmail, SO.SODate

FROM dbo.T_Customer AS C INNER JOIN dbo.T_SalesOrder AS SO ON C.CustID = SO.SOCustID

WHERE (MONTH(SO.SODate) = @month);

Go

Declare @month int

Set @month = '12'

exec Query11 @month
```

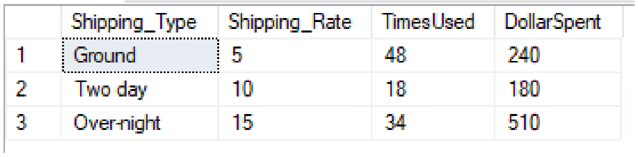


	CustFirstName	CustLastName	CustEmail	SODate
1	Ignacia	Mueller	sit.amet.consectetuer@Aliquamtincidunt.edu	2019-12-11
2	Kylee	Marks	In.condimentum@faucibusidlibero.co.uk	2019-12-25
3	Patrick	Rollins	tortor.at@pellentesqueafacilisis.edu	2020-12-07
4	Aidan	Leach	quis.tristique.ac@aarcu.org	2019-12-05
5	Ivor	Walker	Sed.molestie@ornarefacilisiseget.ca	2019-12-20
6	Cedric	Hayes	egestas.Fusce.aliquet@consequat.edu	2020-12-21
7	Quail	Eaton	Donec.est.Nunc@atsem.net	2020-12-20
8	Rudyard	Frazier	ultrices.sit.amet@blandit.ca	2020-12-27

12.

Number of times used and dollars spent on each shipping vendor and shipping type by vendor.

```
Select S.Shipping_Type, S.Shipping_Rate, COUNT(DO.[DelIEmpID-Ship]) as TimesUsed, S.Shipping_Rate * COUNT(DO.[DelIEmpID-Ship]) as DollarSpent

From T_SalesOrder SO
Join T_DeliveryOut DO
ON SO.SODelIOutID = DO.DelOutID
```



	Shipping_Type	Shipping_Rate	TimesUsed	DollarSpent
1	Ground	5	48	240
2	Two day	10	18	180
3	Over-night	15	34	510

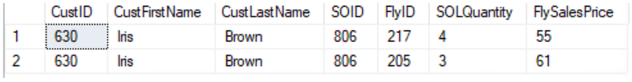
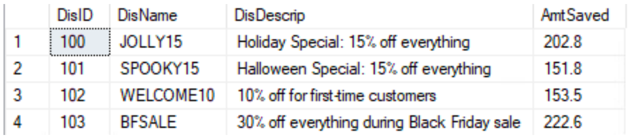
<table>
<tr><td></td><td></td><td>Join T_Shipping S ON
DO.DelOutShippingID =
S.SID

Group By
S.Shipping_Type,
S.Shipping_Rate</td><td></td></tr>
<tr><td>13.</td><td>Invoice lines for a given sales invoice number and given customer name

<em>(Refer to our last assumption with reference to how we track payment info/costs)</em>

.</td><td>SELECT C.CustID,
C.CustFirstName,
C.CustLastName,
SO.SOID, SOL.FlyID,
SOL.SOLQuantity,
FlyCost

FROM T_Customer C
Join T_SalesOrder SO
ON C.CustID =
SO.SOCustID Join
T_SalesOrderLine SOL
ON SO.SOID =
SOL.SOID Join T_Fly F
ON SOL.FlyID = F.FlyID

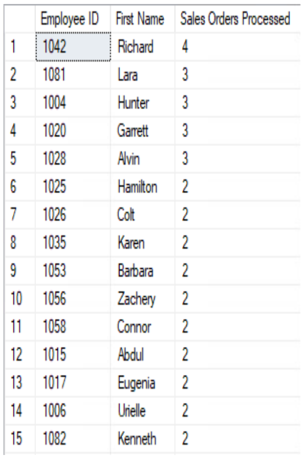
WHERE
C.CustFirstName = 'Iris'
AND C.CustLastName =
'Brown'
AND SO.SOID = ‘806’
AND C.CustID = ‘630’</td><td>
<table>
<tr><th></th><th>CustID</th><th>CustFirstName</th><th>CustLastName</th><th>SOID</th><th>FlyID</th><th>SOLQuantity</th><th>FlySalesPrice</th></tr>
<tr><td>1</td><td>630</td><td>Iris</td><td>Brown</td><td>806</td><td>217</td><td>4</td><td>55</td></tr>
<tr><td>2</td><td>630</td><td>Iris</td><td>Brown</td><td>806</td><td>205</td><td>3</td><td>61</td></tr>
</table></td></tr>
<tr><td>14.</td><td>A number of times a discount was applied to a sales order. List all the information about the discount, the total amount saved by customers that</td><td>SELECT D.DisID,
D.DisName,
D.DisDescrip,
(SUM(F.FlySalesPrice *
SOL.SOLQuantity)*
D.DisRate) as AmtSaved
FROM T_Discount D Join
T_SalesOrder SO ON
D.DisID = SO.SODisID
JOIN T_SalesOrderLine
SOL ON SO.SOID =
SOL.SOID
JOIN T_Fly F ON
SOL.FlyID = F.FlyID</td><td>
<table>
<tr><th></th><th>DisID</th><th>DisName</th><th>DisDescrip</th><th>AmtSaved</th></tr>
<tr><td>1</td><td>100</td><td>JOLLY15</td><td>Holiday Special: 15% off everything</td><td>202.8</td></tr>
<tr><td>2</td><td>101</td><td>SPOOKY15</td><td>Halloween Special: 15% off everything</td><td>151.8</td></tr>
<tr><td>3</td><td>102</td><td>WELCOME10</td><td>10% off for first-time customers</td><td>153.5</td></tr>
<tr><td>4</td><td>103</td><td>BFSALE</td><td>30% off everything during Black Friday sale</td><td>222.6</td></tr>
</table></td></tr>
</table>

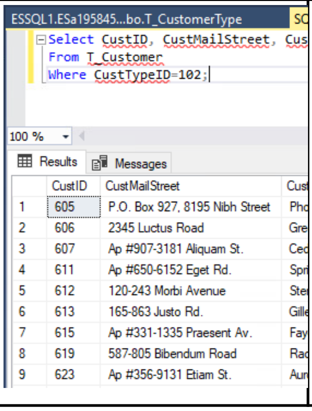
	used the discount.	GROUP BY D.DisID, D.DisName, D.DisDescrip, D.DisRate	

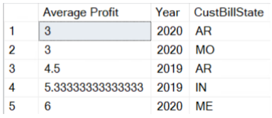
Three Additional Queries

In the section below Big Data has created three additional queries that we felt would be useful for Elysian Fly Company. These queries were made by evaluating the needs of Elysian Fly Company and seeing what best fit them. These three unique queries should help Elysian Fly Company evaluate how they can be more effective and efficient. Below, the question, why it's important, the SQL code, the output, and a recap of the findings are displayed.

Query #	Question	Why is this important	SQL	Partial Output	Recap of Findings

1	Which employees are most involved with processing SalesOrder?
	This query captures information from when the employee first began work at Elysian.
		We need to know this so we can see which employees are the most active within the organization when it comes to getting SalesOrders through our system.	SELECT SO.SOEmpID AS [Employee ID], Emp.EmpFirstName AS [First Name], COUNT(SO.SOID) AS [Sales Orders Processed]
			FROM T_Employee Emp, T_SalesOrder as SO
			WHERE Emp.EmpID = SO.SOEmpID
			GROUP BY SO.SOEmpID, Emp.EmpFirstName
			ORDER BY [Sales Orders Processed] desc;
				
				Employee ID, First Name, Sales Orders Processed
				1 1042 Richard 4
				2 1081 Lara 3
				3 1004 Hunter 3
				4 1020 Garrett 3
				5 1028 Alvin 3
				6 1025 Hamilton 2
				7 1026 Colt 2
				8 1035 Karen 2
				9 1053 Barbara 2
				10 1056 Zachery 2
				11 1058 Connor 2
				12 1015 Abdul 2
				13 1017 Eugenia 2
				14 1006 Urielle 2
				15 1082 Kenneth 2
					We found that most of the employees seem to process a fairly equivalent amount of Sales Orders.

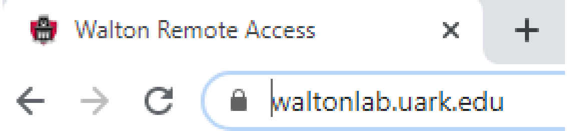
2	Who is a local customer that we can incentivize to go on more guided trips?	This exquisite code exists to inform local patrons of upcoming guided trips as well as frequent shopper discounts.	Select CustID, CustMailStreet, CustMailCity, CustMailState, CustMailZip, CustEmail, CustFirstName, CustLastName From T_Customer Where CustTypeID=102;	 ESSQL1.ESa195845...bo.T_CustomerType Select CustID, CustMailStreet, Cus From T_Customer Where CustTypeID=102; 100 % Results Messages CustID, CustMailStreet, Cust 1 605 P.O. Box 927, 8195 Nibh Street Pho 2 606 2345 Luctus Road Gre 3 607 Ap #907-3181 Aliquam St. Ced 4 611 Ap #650-6152 Eget Rd. Spri 5 612 120-243 Morbi Avenue Ste 6 613 165-863 Justo Rd. Gille 7 615 Ap #331-1335 Praesent Av. Fay 8 619 587-805 Bibendum Road Rad 9 623 Ap #356-9131 Etiam St. Aur	There were a significant number of local clients.

3	What is the average profit made from the sale of Flies each year by customer state and which states do we need to target more?	This output is important because it can show Elysian what states are bringing in the most profit, and which ones they need to reach more.	Select AVG(FlySalesPrice-FlyCost) as "Average Profit", Year(SODate) as "Year", CustBillState From T_Customer C JOIN T_SalesOrder SO ON C.CustID=SO.SOCustID JOIN T_SalesOrderLine SOL ON SO.SOID=SOL.SOID JOIN T_Fly F ON SOL.FlyID=F.FlyID Group By Year(SODate), CustBillState Order By "Average Profit" asc	 Average Profit | Year | CustBillState 1 | 3 | 2020 | AR 2 | 3 | 2020 | MO 3 | 4.5 | 2019 | AR 4 | 5.33333333333333 | 2019 | IN 5 | 6 | 2020 | ME	The top 5 lowest profiting states were shown, which can be alleviated by active marketing.

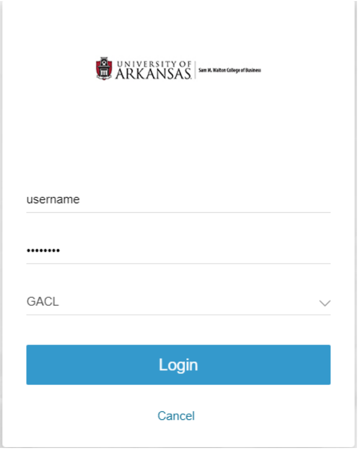
User Documentation

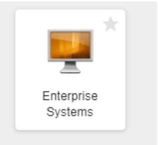
In this section, we will detail to our end-user how to access their new database, and secondly, the step-by-step instructions to view relevant queries/views to find information the client has requested. The instructions will encompass the entire process, from loading up the Virtual Machine all the way through to viewing the results of queries created within the database.

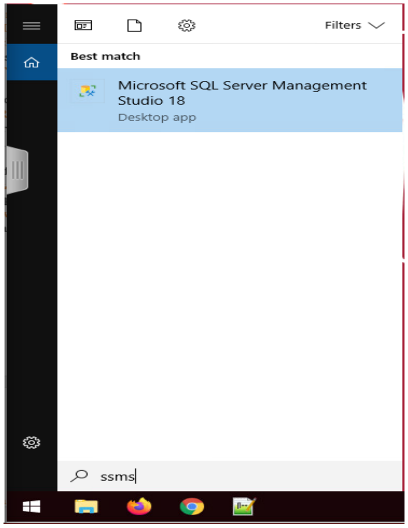
First, you need to open a browser and go to the website “ **https://waltonlab.uark.edu** “.

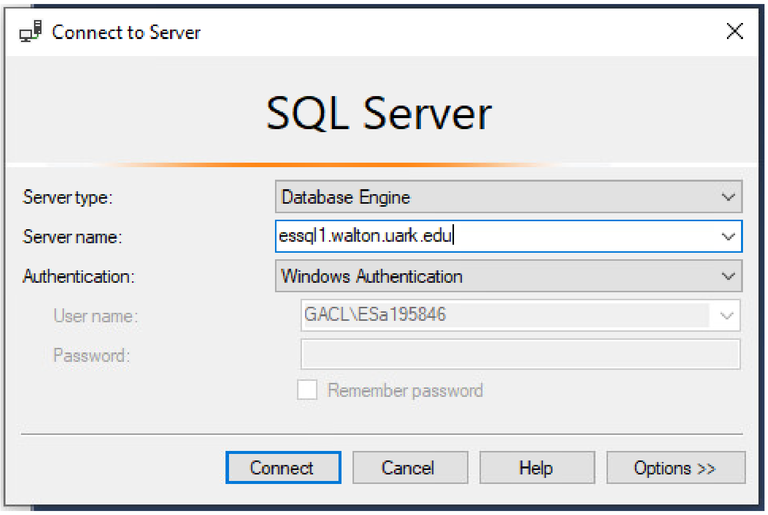


When the website opens, select the “**Access Virtual Desktop With Web Browser**” option, this is located on the right side of the screen. Then click **“accept”**, this will prompt you to enter your credentials. You need to make sure that the domain is **“GACL”**. Click **“Login.”** Once you are logged in, click on the “**Enterprise Systems**” icon. This will take you to a virtual desktop. From here, go to the **Windows Start** button in the lower left corner of the screen and click on it. Scroll to find the **search tool** and type in “**SSMS**”, the program titled **“Microsoft SQL Server Management Studio 18”** will pop up, click open.









Connect to Server

SQL Server

Server type: Database Engine

Server name: essql1.walton.uark.edu

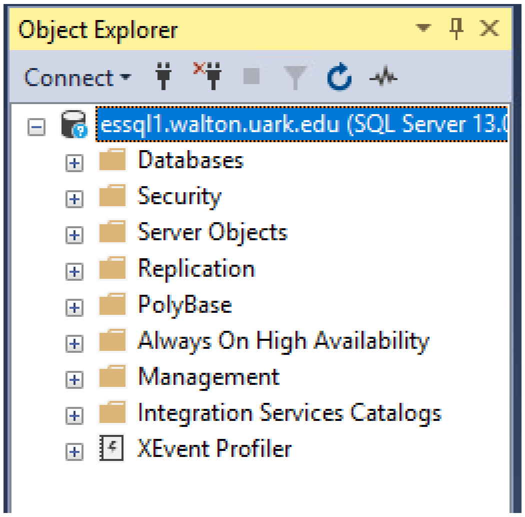
Authentication: Windows Authentication

User name: GACL\ESa195846

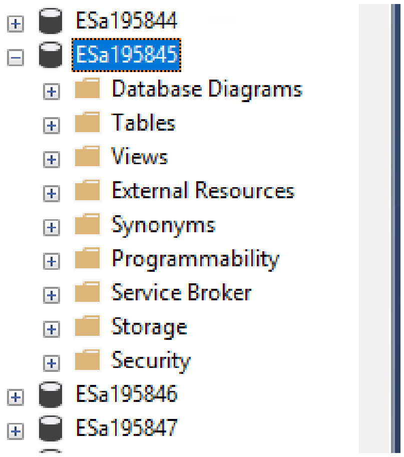
Password:

□ Remember password

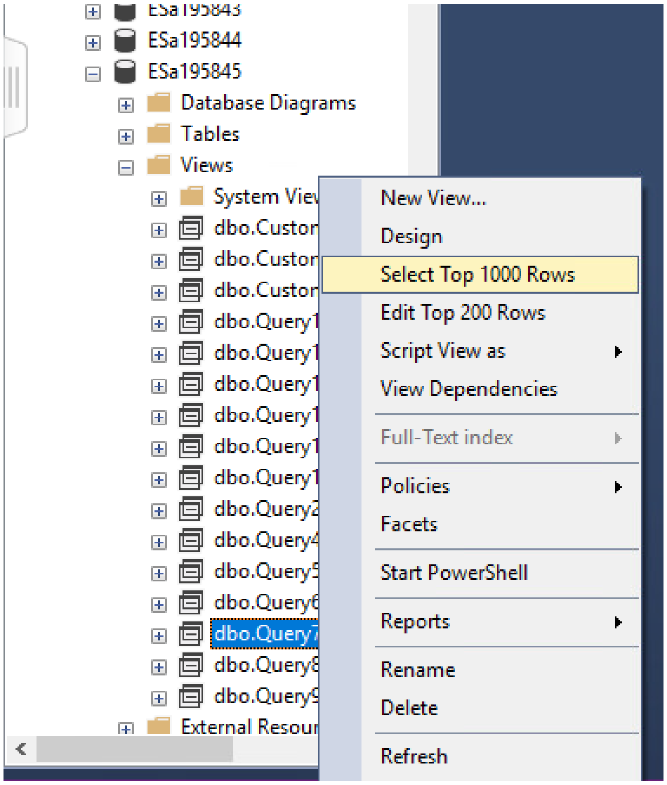
Once that opens, a pop up will appear prompting you to enter a server name, enter server name "**essql1.walton.uark.edu**" and click connect.
Once connected, you are on the correct server. On the left side of the screen, there is a pop up

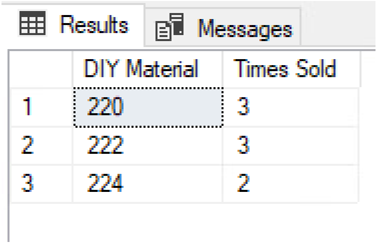


with several different folders, expand the **Databases** folder. Next, scroll down on the list of databases and find **ESa195845**, once found click the plus button to expand.



From here you can look at the implemented physical design under the **Database Diagrams** folder, each table under the **Tables** folder, and all queries under the **Views** folder. In order to access a View, right-click on the View corresponding to the query you wish to execute. Click the "**Select Top 1000 Rows**" option to display the results in a column/row display form.





	DIY Material	Times Sold
1	220	3
2	222	3
3	224	2

What We Learned Throughout This Process

Throughout this process Big Data has successfully overcome several obstacles together. As a team, Big Data learned how to create solutions together, manage our time, and ensure that all of the clients needs were met. While working on this project, Big Data became engrossed in ensuring that every detail was perfect so that we could output our best work, however that sometimes developed into overthinking tasks that otherwise would have been simple.
As a team we also learned to take a step back and make sure that we were solving problems in the most efficient way while utilizing each team member's strengths and working as a team. Written below are a few lessons learned throughout this experience from each team member, as we would like to reflect on our accomplishments and what we learned from them.

Member Name:	**What you learned:**
Olivia Long	While working on this extensive project, I have learned several things about database creation and working with a team. At first I was not completely solid on ERD work, but as we got into creating our own my confidence increased. A large part of this can be attributed to the help from my team, we all worked together and helped explain different things to each other, increasing all of our knowledge

	together. I think for me that was the biggest take away of the project, when working as a team we built on each member's strengths and helped each other learn, making all of us much more knowledgeable by the end of the project. A personal accomplishment I was proud of was being able to execute my queries after going through much trial and error. I was also proud that I was able to create my own custom query that worked and would be helpful to the client. Another personal accomplishment I had was being able to explain what we were doing in words. I tend to struggle with explaining things, but as I wrote about denormalization, referential integrity, and physical design, I became even more confident in my knowledge. Overall this project has significantly increased my knowledge in databases and has taught me how to work effectively with a team while being successful.
Marshall Mann	This project really felt like a capstone to this course. It was a practical application of what we learned in class. Reflecting on our journey at the end of it, I think about how impossible it would seem that we did all of this. If I had understood the magnitude of this project at the beginning I wouldn't have believed it. I liked how we got to apply everything we learned in class. I learned how to create an ERD and that sometimes I have a tendency to go too far into detail. I learned that large projects can be broken up into small chunks. This project would have seemed overwhelming without periodic milestones. The milestones really helped to make it manageable. I learned that communication and setting micro deadlines is important when working as a team. Meeting briefly and assigning parts helps keep the project moving along. I enjoyed using Microsoft SQL Server and actually implementing our database. Being able to run queries was very satisfying. I found generating all of the data for the tables to be interesting because sometimes I could use the generated data website but for things like lures I researched actual popular lures to add to our product table. Generating the data was challenging but rewarding because you really had to use your head.
Evan Langenberg	Finally coming to the end of this project, I am incredibly proud with how we worked as a team to complete all the tasks that were placed before us. Even at individual levels of competency that varied depending on the unit we were in, we all managed to leverage our strengths and be incredibly vulnerable to improve and move forward. We encountered lots of roadblocks in Milestone I, overthinking and overanalyzing our requirements and making our workload way too intense. However, after refining our processes and having informal group roles being established, our team evolved into a productivity powerhouse. We all are incredibly thankful that we were paired together, as none of us have ever had a group in our careers as students that has been this successful and easy-going. For individual achievements, I believe I grew exponentially as not only a leader as I became more confident in my understanding and mastery of the course material, but as a future member of the workforce. This project was as close a simulation to real-life, remote-working scenarios that are more common now post-COVID than any other era in history. Moreover, I improved at my ability to break down large goals into attainable, measurable steps, and I also improved my ability to delegate. This project had many moving parts, with instructions and data being referenced from multiple documents from multiple sources, and one of the keys to our success

	was properly sifting through all the information in front of us to derive the tasks at hand. Finally, this project was a massive boost to my interpersonal skills. From interacting with the professor regularly to receive guidance and direction, and working with my colleagues in Big Data, my ability to liaise and collaborate with others to reach a common goal improved dramatically during the course of this project.
Karen Li	I have learned a lot while working on this project this semester. To me, this class was more than a class where you would only sit down and take notes. This class and project has provided me experience to simulate what real life would look like if I was working under a company. While working on the project, I have definitely improved my communication and team working skills. Without a team, this project wouldn't have been as successful as it is right now and it would have been more stressful if I was working on it alone. Thus, I am very grateful for our team and each of the members. In milestone 1, we faced many challenges when we were building our ERD. There were instances where we were over-complicating things when it could've been easily fixed by adding an attribute or creating a reference table. A good lesson I took away from that is that it is okay to take a step back in our thought process and ask for help. Another skill I have improved on is time management. This project would have been impossible to complete if it was done in one sitting. I have learned that setting weekly deadlines and distributing equal tasks among our team members has helped our project move forward and feel a lot less intimidating. The project has also helped me understand the course material better while applying what we learned in the classroom. One last skill I have gained while working on this project is using the SQL Server Management Studio. Prior to this class, I had zero experience with this software. I liked how easy it was to create and assemble your database and implement it.
Quannah Lindsey	This process has been an eye opening experience and I have learned so much. During milestone 1 I learned that it is important to keep the ERD as easy and streamlined as possible and it will make things easier in the long run. When the ERD was constructed it was neat to see how all of the tables fit together and were able to be linked with their primary and foreign keys as well as the reference tables. During milestone 2 normalization I struggled because the concept of normalization can seem difficult and it really is until you have actually worked through the process. I enjoyed getting to see everything broken down and written out to see how everything was again connected. I learned during the normalization process that it is important to stay organized so that it is easier to write everything down. Milestone 3 has been the most enjoyable for me because you actually get to see all of the data come together and get to see how everything is working together and linked physically. I learned how to troubleshoot our queries until I was able to get the results that the company was asking for and although it was stressful when they were not working it was exciting when I figured it out and was able to move onto the next query. The process overall has taught me life skills in the field that I wouldn't have gotten from other classes. This process has taught me patience and a different way of thinking about things and how data tables work together to create a whole process. This team has been instrumental in my success in this process and I can honestly say I wouldn't have it any other way. As a team we worked very

	well together and we all had a helpful attitude when someone was struggling or needed help. We struggled through some of the processes, but we struggled as a team and my only negative is that I cannot keep this team throughout the rest of my college career.

Appendix

Team Contract

Team Name: **Big Data**
Team Motto: **"Don't compromise on excellence."**

Team Members

Name	Email	Phone	Strengths	Availability to Meet
Marshall Mann	marshall.w.mann-1@ou.edu	405-420-6955	Organization, focus, [I am often away from my phone and email when studying. I will not be able to respond immediately and thus should be contacted for group meetings at least a day in advance.]	Monday 1:40 - 4:30 pm, Tuesday 2:45 - 4:30 pm, Wednesday 1:30-3:45, other times by appointment.
Quannah Lindsey	quannah.j.lindsey-1@ou.edu	580-399-8581	Detail oriented, time management	Most evenings, weekends (I work during the day when i'm not in class)
Evan Langenberg	evan.m.langenberg-1@ou.edu	281-799-2815	Time management, being able to empathize and listen, and the desire to learn	Pretty much any day after 5, and the weekends
Karen Li	karen.l.li@ou.edu	918-706-3705	Organization, Patience, Responsibility, Discipline	Mon 2-6pm, Tues 1-5pm, Wed 2-6pm, Thurs 1-5pm Fri 2-5pm
Olivia Long	olivialong4@ou.edu	972-567-6163	Responsibility, work well with others, staying on task	Everyday after 5 except for Thursdays and most weekends

Team Expectations for the confidential peer evaluation :

Clear communication of roles within the team and responsibility for your section, meeting deadlines, doing good quality work, and being on top of communications through Teams or otherwise.

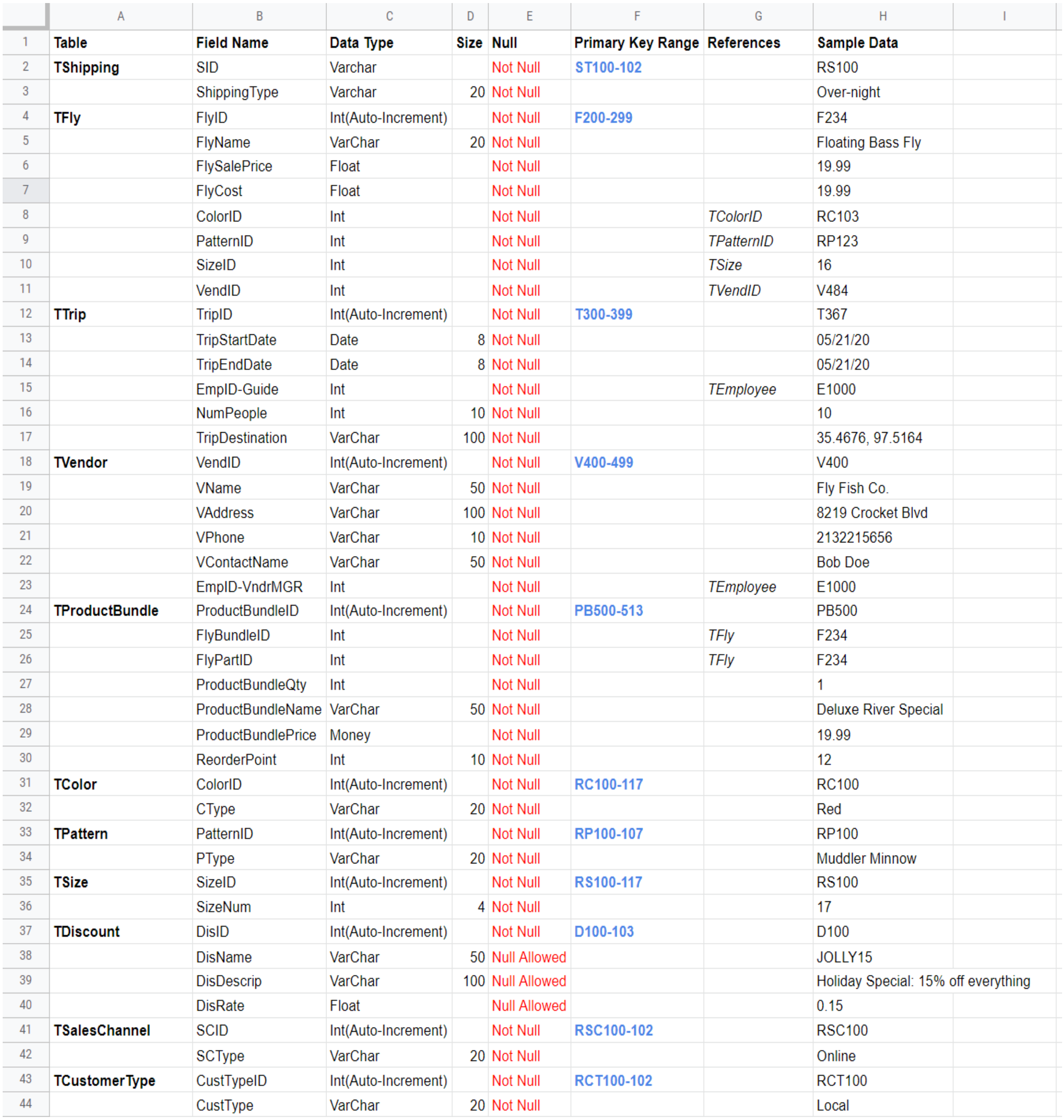
The behavior for which points will be deducted on the confidential peer evaluation:

Not finishing your assigned section, bad/absent communication, submitting poor quality work, being disrespectful or being rude to fellow team members.

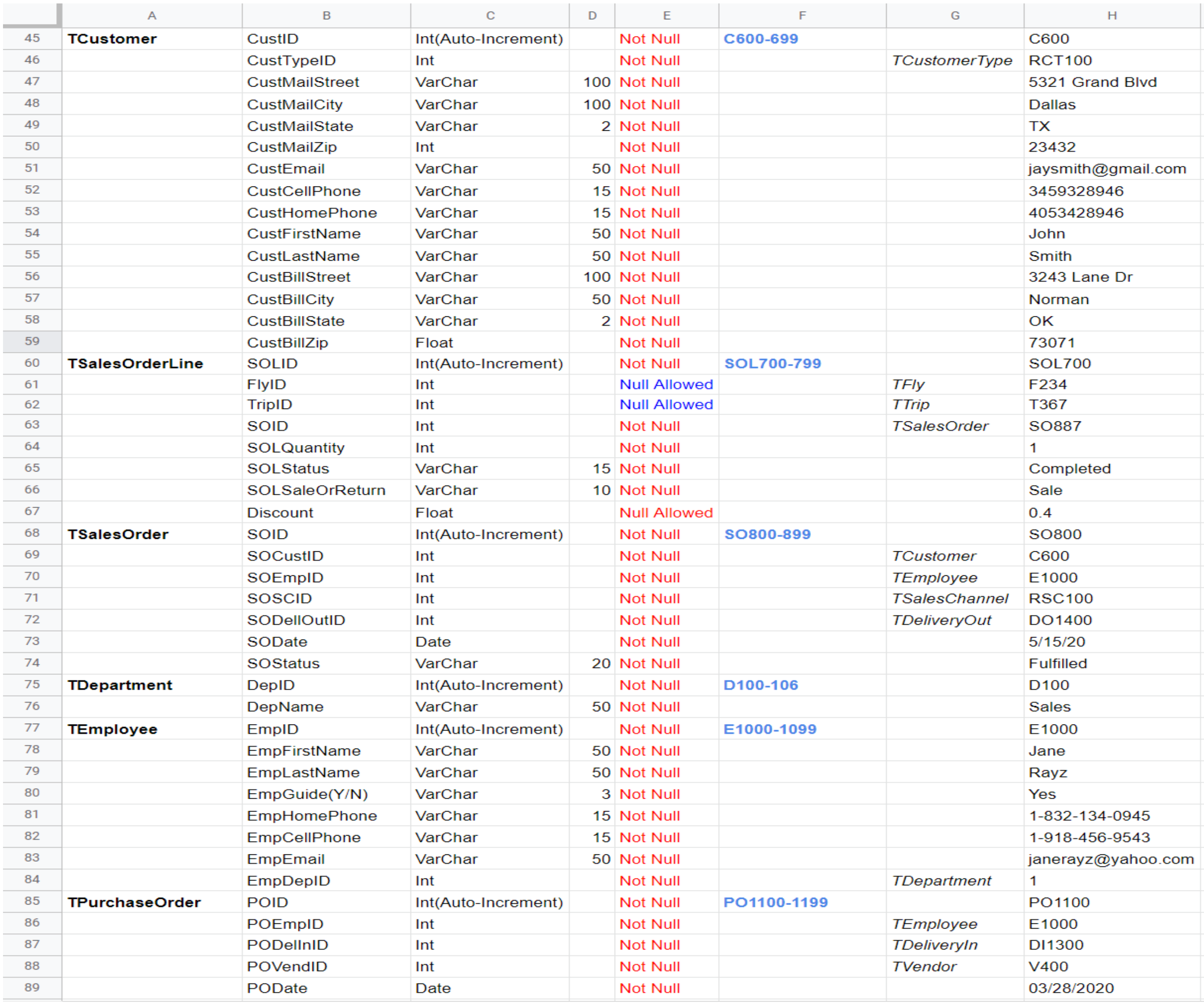
Presentation Date Preferences (Rank Order Available Dates; make sure you list dates that absolutely don't work for your team):

October 23, November 9, November 23

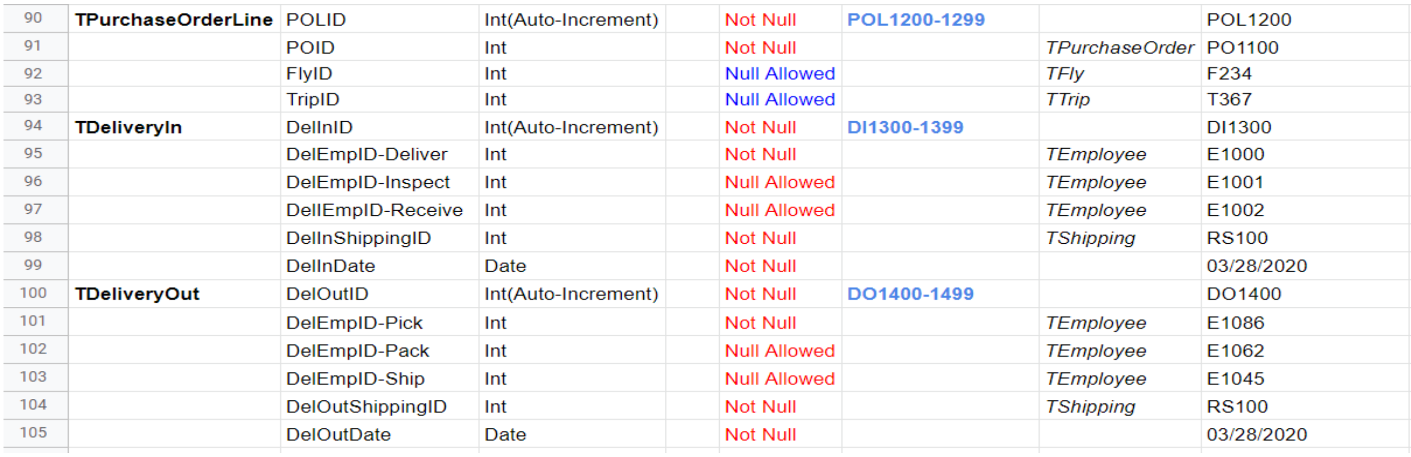
Data Dictionary Model



	A	B	C	D	E	F	G	H	I
1	**Table**	**Field Name**	**Data Type**	**Size**	**Null**	**Primary Key Range**	**References**	**Sample Data**	
2	**TShipping**	SID	Varchar		Not Null	**ST100-102**		RS100	
3		ShippingType	Varchar	20	Not Null			Over-night	
4	**TFly**	FlyID	Int(Auto-Increment)		Not Null	**F200-299**		F234	
5		FlyName	VarChar	20	Not Null			Floating Bass Fly	
6		FlySalePrice	Float		Not Null			19.99	
7		FlyCost	Float		Not Null			19.99	
8		ColorID	Int		Not Null		*TColorID*	RC103	
9		PatternID	Int		Not Null		*TPatternID*	RP123	
10		SizeID	Int		Not Null		*TSize*	16	
11		VendID	Int		Not Null		*TVendID*	V484	
12	**TTrip**	TripID	Int(Auto-Increment)		Not Null	**T300-399**		T367	
13		TripStartDate	Date	8	Not Null			05/21/20	
14		TripEndDate	Date	8	Not Null			05/21/20	
15		EmpID-Guide	Int		Not Null		*TEmployee*	E1000	
16		NumPeople	Int	10	Not Null			10	
17		TripDestination	VarChar	100	Not Null			35.4676, 97.5164	
18	**TVendor**	VendID	Int(Auto-Increment)		Not Null	**V400-499**		V400	
19		VName	VarChar	50	Not Null			Fly Fish Co.	
20		VAddress	VarChar	100	Not Null			8219 Crocket Blvd	
21		VPhone	VarChar	10	Not Null			2132215656	
22		VContactName	VarChar	50	Not Null			Bob Doe	
23		EmpID-VndrMGR	Int		Not Null		*TEmployee*	E1000	
24	**TProductBundle**	ProductBundleID	Int(Auto-Increment)		Not Null	**PB500-513**		PB500	
25		FlyBundleID	Int		Not Null		*TFly*	F234	
26		FlyPartID	Int		Not Null		*TFly*	F234	
27		ProductBundleQty	Int		Not Null			1	
28		ProductBundleName	VarChar	50	Not Null			Deluxe River Special	
29		ProductBundlePrice	Money		Not Null			19.99	
30		ReorderPoint	Int	10	Not Null			12	
31	**TColor**	ColorID	Int(Auto-Increment)		Not Null	**RC100-117**		RC100	
32		CType	VarChar	20	Not Null			Red	
33	**TPattern**	PatternID	Int(Auto-Increment)		Not Null	**RP100-107**		RP100	
34		PType	VarChar	20	Not Null			Muddler Minnow	
35	**TSize**	SizeID	Int(Auto-Increment)		Not Null	**RS100-117**		RS100	
36		SizeNum	Int	4	Not Null			17	
37	**TDiscount**	DisID	Int(Auto-Increment)		Not Null	**D100-103**		D100	
38		DisName	VarChar	50	Null Allowed			JOLLY15	
39		DisDescrip	VarChar	100	Null Allowed			Holiday Special: 15% off everything	
40		DisRate	Float		Null Allowed			0.15	
41	**TSalesChannel**	SCID	Int(Auto-Increment)		Not Null	**RSC100-102**		RSC100	
42		SCType	VarChar	20	Not Null			Online	
43	**TCustomerType**	CustTypeID	Int(Auto-Increment)		Not Null	**RCT100-102**		RCT100	
44		CustType	VarChar	20	Not Null			Local	



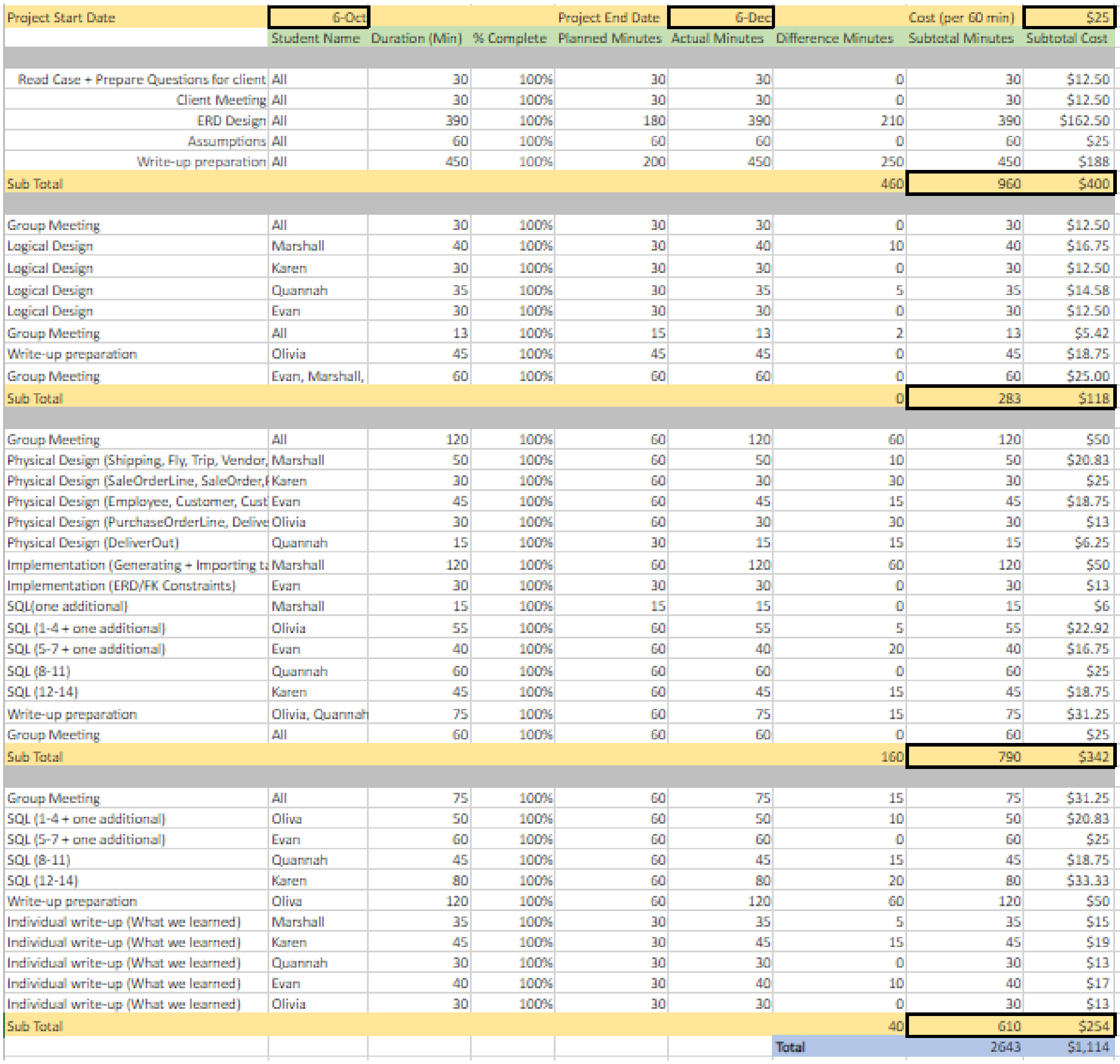
	A	B	C	D	E	F	G	H
45	TCustomer	CustID	Int(Auto-Increment)		Not Null	C600-699		C600
46		CustTypeID	Int		Not Null		TCustomerType	RCT100
47		CustMailStreet	VarChar	100	Not Null			5321 Grand Blvd
48		CustMailCity	VarChar	100	Not Null			Dallas
49		CustMailState	VarChar	2	Not Null			TX
50		CustMailZip	Int		Not Null			23432
51		CustEmail	VarChar	50	Not Null			jaysmith@gmail.com
52		CustCellPhone	VarChar	15	Not Null			3459328946
53		CustHomePhone	VarChar	15	Not Null			4053428946
54		CustFirstName	VarChar	50	Not Null			John
55		CustLastName	VarChar	50	Not Null			Smith
56		CustBillStreet	VarChar	100	Not Null			3243 Lane Dr
57		CustBillCity	VarChar	50	Not Null			Norman
58		CustBillState	VarChar	2	Not Null			OK
59		CustBillZip	Float		Not Null			73071
60	TSalesOrderLine	SOLID	Int(Auto-Increment)		Not Null	SOL700-799		SOL700
61		FlyID	Int		Null Allowed		TFly	F234
62		TripID	Int		Null Allowed		TTrip	T367
63		SOID	Int		Not Null		TSalesOrder	SO887
64		SOLQuantity	Int		Not Null			1
65		SOLStatus	VarChar	15	Not Null			Completed
66		SOLSaleOrReturn	VarChar	10	Not Null			Sale
67		Discount	Float		Null Allowed			0.4
68	TSalesOrder	SOID	Int(Auto-Increment)		Not Null	SO800-899		SO800
69		SOCustID	Int		Not Null		TCustomer	C600
70		SOEmpID	Int		Not Null		TEmployee	E1000
71		SOSCID	Int		Not Null		TSalesChannel	RSC100
72		SODelIOutID	Int		Not Null		TDeliveryOut	DO1400
73		SODate	Date		Not Null			5/15/20
74		SOStatus	VarChar	20	Not Null			Fulfilled
75	TDepartment	DepID	Int(Auto-Increment)		Not Null	D100-106		D100
76		DepName	VarChar	50	Not Null			Sales
77	TEmployee	EmpID	Int(Auto-Increment)		Not Null	E1000-1099		E1000
78		EmpFirstName	VarChar	50	Not Null			Jane
79		EmpLastName	VarChar	50	Not Null			Rayz
80		EmpGuide(Y/N)	VarChar	3	Not Null			Yes
81		EmpHomePhone	VarChar	15	Not Null			1-832-134-0945
82		EmpCellPhone	VarChar	15	Not Null			1-918-456-9543
83		EmpEmail	VarChar	50	Not Null			janerayz@yahoo.com
84		EmpDepID	Int		Not Null		TDepartment	1
85	TPurchaseOrder	POID	Int(Auto-Increment)		Not Null	PO1100-1199		PO1100
86		POEmpID	Int		Not Null		TEmployee	E1000
87		PODelInID	Int		Not Null		TDeliveryIn	DI1300
88		POVendID	Int		Not Null		TVendor	V400
89		PODate	Date		Not Null			03/28/2020



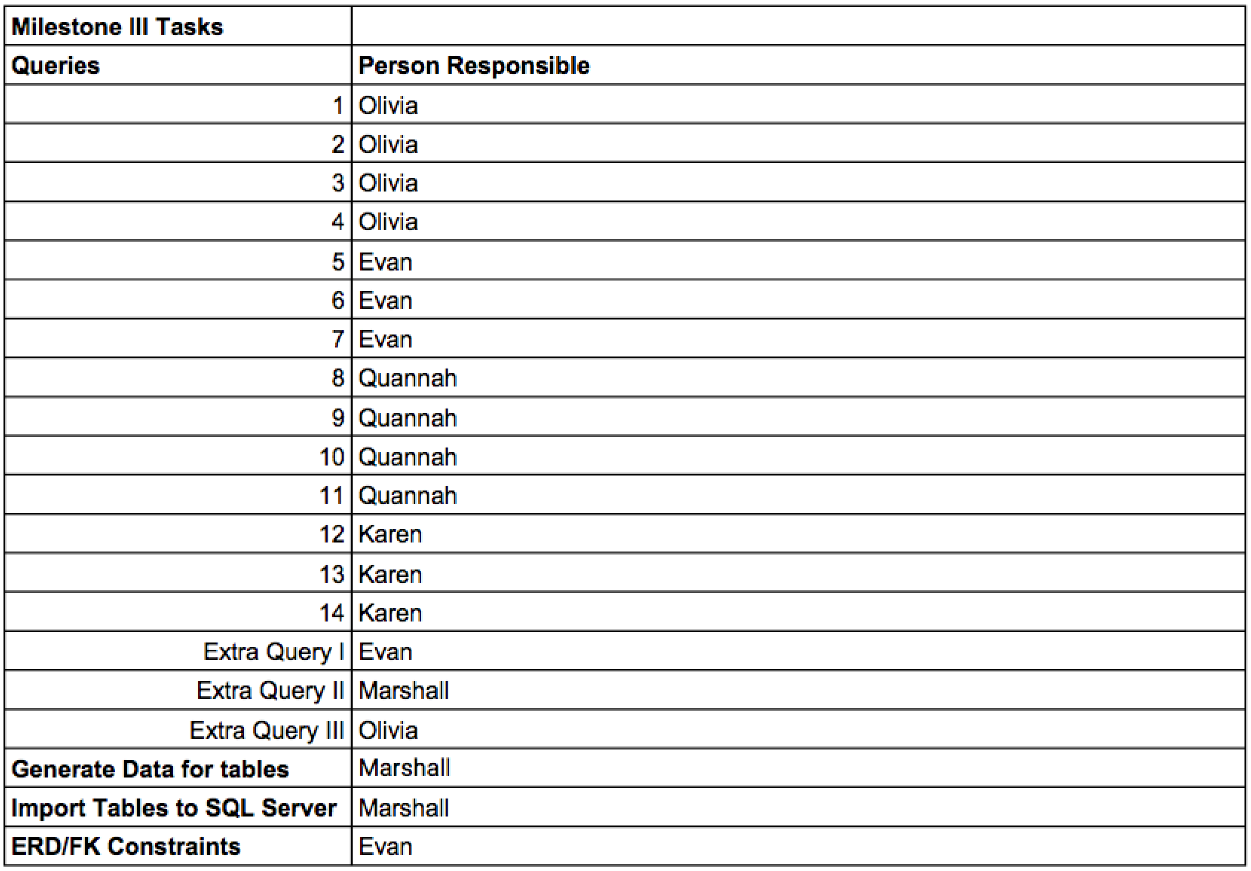
90	TPurchaseOrderLine	POLID	Int(Auto-Increment)		Not Null	POL1200-1299		POL1200
91		POID	Int		Not Null		TPurchaseOrder	PO1100
92		FlyID	Int		Null Allowed		TFly	F234
93		TripID	Int		Null Allowed		TTrip	T367
94	TDeliveryIn	DelInID	Int(Auto-Increment)		Not Null	DI1300-1399		DI1300
95		DelEmpID-Deliver	Int		Not Null		TEmployee	E1000
96		DelEmpID-Inspect	Int		Null Allowed		TEmployee	E1001
97		DellEmpID-Receive	Int		Null Allowed		TEmployee	E1002
98		DelInShippingID	Int		Not Null		TShipping	RS100
99		DelInDate	Date		Not Null			03/28/2020
100	TDeliveryOut	DelOutID	Int(Auto-Increment)		Not Null	DO1400-1499		DO1400
101		DelEmpID-Pick	Int		Not Null		TEmployee	E1086
102		DelEmpID-Pack	Int		Null Allowed		TEmployee	E1062
103		DelEmpID-Ship	Int		Null Allowed		TEmployee	E1045
104		DelOutShippingID	Int		Not Null		TShipping	RS100
105		DelOutDate	Date		Not Null			03/28/2020

Project Management

The project management tool helped our team ensure that everyone was doing their part and how much we should charge to Elysian Fly Company for our work. We learned that as we got deeply invested in this project, we were spending large amounts of time working on it, but that we needed to only add minutes spent actually creating and solving solutions. This helped us better manage our time in meetings, and kept our price down so that we would be a more appealing choice to Elysian Fly Company.



Project Start Date	6-Oct			Project End Date	6-Dec		Cost (per 60 min)	$25
	Student Name	Duration (Min)	% Complete	Planned Minutes	Actual Minutes	Difference Minutes	Subtotal Minutes	Subtotal Cost
Read Case + Prepare Questions for client	All	30	100%	30	30	0	30	$12.50
Client Meeting	All	30	100%	30	30	0	30	$12.50
ERD Design	All	390	100%	180	390	210	390	$162.50
Assumptions	All	60	100%	60	60	0	60	$25
Write-up preparation	All	450	100%	200	450	250	450	$188
Sub Total						460	960	$400
Group Meeting	All	30	100%	30	30	0	30	$12.50
Logical Design	Marshall	40	100%	30	40	10	40	$16.75
Logical Design	Karen	30	100%	30	30	0	30	$12.50
Logical Design	Quannah	35	100%	30	35	5	35	$14.58
Logical Design	Evan	30	100%	30	30	0	30	$12.50
Group Meeting	All	13	100%	15	13	2	13	$5.42
Write-up preparation	Olivia	45	100%	45	45	0	45	$18.75
Group Meeting	Evan, Marshall,	60	100%	60	60	0	60	$25.00
Sub Total						0	283	$118
Group Meeting	All	120	100%	60	120	60	120	$50
Physical Design (Shipping, Fly, Trip, Vendor,	Marshall	50	100%	60	50	10	50	$20.83
Physical Design (SaleOrderLine, SaleOrder, F	Karen	30	100%	60	30	30	30	$25
Physical Design (Employee, Customer, Cust	Evan	45	100%	60	45	15	45	$18.75
Physical Design (PurchaseOrderLine, Delive	Olivia	30	100%	60	30	30	30	$13
Physical Design (DeliverOut)	Quannah	15	100%	30	15	15	15	$6.25
Implementation (Generating + Importing ta	Marshall	120	100%	60	120	60	120	$50
Implementation (ERD/FK Constraints)	Evan	30	100%	30	30	0	30	$13
SQL(one additional)	Marshall	15	100%	15	15	0	15	$6
SQL (1-4 + one additional)	Olivia	55	100%	60	55	5	55	$22.92
SQL (5-7 + one additional)	Evan	40	100%	60	40	20	40	$16.75
SQL (8-11)	Quannah	60	100%	60	60	0	60	$25
SQL (12-14)	Karen	45	100%	60	45	15	45	$18.75
Write-up preparation	Olivia, Quannah	75	100%	60	75	15	75	$31.25
Group Meeting	All	60	100%	60	60	0	60	$25
Sub Total						160	790	$342
Group Meeting	All	75	100%	60	75	15	75	$31.25
SQL (1-4 + one additional)	Oliva	50	100%	60	50	10	50	$20.83
SQL (5-7 + one additional)	Evan	60	100%	60	60	0	60	$25
SQL (8-11)	Quannah	45	100%	60	45	15	45	$18.75
SQL (12-14)	Karen	80	100%	60	80	20	80	$33.33
Write-up preparation	Oliva	120	100%	60	120	60	120	$50
Individual write-up (What we learned)	Marshall	35	100%	30	35	5	35	$15
Individual write-up (What we learned)	Karen	45	100%	30	45	15	45	$19
Individual write-up (What we learned)	Quannah	30	100%	30	30	0	30	$13
Individual write-up (What we learned)	Evan	40	100%	30	40	10	40	$17
Individual write-up (What we learned)	Olivia	30	100%	30	30	0	30	$13
Sub Total						40	610	$254
						Total	2643	$1,114



Milestone III Tasks	
Queries	**Person Responsible**
1	Olivia
2	Olivia
3	Olivia
4	Olivia
5	Evan
6	Evan
7	Evan
8	Quannah
9	Quannah
10	Quannah
11	Quannah
12	Karen
13	Karen
14	Karen
Extra Query I	Evan
Extra Query II	Marshall
Extra Query III	Olivia
Generate Data for tables	Marshall
Import Tables to SQL Server	Marshall
ERD/FK Constraints	Evan