

University of Oklahoma

Mozart

Hillary Neaves, Khady Ndoye, Lam Nguyen, Sahrin Moytree, Sara Mullins

MIS 3353 - Database Management

SQL server: ESa195725

Frederic Daugherty

2020

Executive Summary

Mozart is a database design and development firm that offers a wide range of IT services to its clients to provide effective data management and storage. For more than 16 years, Mozart has been delivering top-notch curations of tailor-made databases and consultation for unique businesses all over the global community. We provide databases that carefully shelf a company's data in a secure and reliable environment. The database can be optionally integrated with other data sources to avoid manual input, which allows companies to use data in a fast and convenient way. We strive to deliver exceptional service and maintain a strong connection with our customers. Mozart has evolved into a full database development firm which creates a wide range of digital products for its clientele. We are famous for our proactive approach, our expert technical professionals prioritize client satisfaction, success, and happiness above all.

Elysian Fly Company, a growing retailer of fly-fishing merchandise, is searching for a database that allows them to increase profitability and enable further growth. The company is seeking to better track and maintain records of inventory, customers, employees, and equipment. Mozart seeks an opportunity from Elysian Fly Company to curate a custom database software for each entity needed. The firm promises to deliver a clean and secure data storage and seamlessly integrate the company's existing database with state-of-the-art sources. Through the three stages of a database building process conceptual, logical, and physical design, the firm will choose only an efficient design to ensure the most beneficial outcome.

The conceptual design phase consists of a carefully curated Entity Relationship Diagram built specifically for Elysian Fly Company, which portrays two important factors of the database: major entities within the project, and the inter-relationships among these entities. By visualizing these relationships, the designers of the database can easily make the connections between each category of information needed. The logical design step is implemented because the entities are converted into relational tables, which will manage to store data regarding a company's entities in a non-redundant manner. The stage will also be placing foreign keys so that all relationships among the entities will be supported. The final step of database design, implementation, includes a series of steps leading to operational information system that involves creating database definitions, testing the system, developing operational procedure and documentation, and populating data. The cost of the project varies by the extra services needed by Team Mozart. To build an efficient database, which has been certified by our team members, Elysian Fly Company can expect a total cost ranging anywhere from \$10,000- \$15,000. The database will take an estimated three months to build and proofread. The company can stay rest assured knowing their data is in the secure hands of Mozart's top five excellent members.

Contents

| | |
|--|-----------|
| Executive Summary | 2 |
| Get to Know the Team: Mozart..... | 5 |
| Conceptual Design | 7 |
| The Client Meeting | 7 |
| Q&A During the Meeting & Information We Learned | 7 |
| Here you will list each question individually, followed by the answer you received during the interview | 7 |
| Significant Assumptions | 8 |
| What is an ERD? Why is it necessary? | 8 |
| Business Cycles Used | 8 |
| ERD Created | 10 |
| Logical Design | 14 |
| What is Normalization? What purpose does it serve?..... | 14 |
| Define normalization and the importance of including it in this project. Be specific..... | 14 |
| Normalized Relations..... | 14 |
| What differences are there between the ERD and the normalized relations? How do these differences help in moving from conceptual design to implementation?..... | 17 |
| What is a Referential Integrity Constraint? Why are they necessary?..... | 18 |
| Physical Design and Implementation | 19 |
| What is a Data Dictionary? | 30 |
| Denormalization..... | 30 |
| Implemented Physical Design | 31 |
| Please include a screenshot of your implemented physical ERD..... | 31 |
| Once you are done, delete this box. | 31 |
| Challenges Faced/Addressed During Implementation | 32 |
| Strengths and Weaknesses Encountered During Implementation..... | 33 |
| Specific SQL Statements Requested..... | 33 |
| Three Additional Queries | 38 |

| | |
|---|-----------|
| User Documentation | 40 |
| What We Learned Throughout This Process..... | 52 |
| Appendix..... | 54 |
| Team Contract | 54 |
| Data Dictionary Model | 55 |
| Project Management..... | 58 |

Get to Know the Team: Mozart



Sahrin Moytree

Management Information Systems and Human Relations - Minor: Management

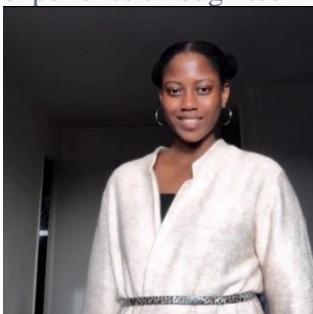
As a former student pilot, Moytree is passionate about connecting technology and humans in a creative way.



Sara Mullins

Management Information Systems and International Business - Minor: French

Mullins is a junior from Kansas City, Missouri. Her passion lies in cyber security and improving user experience through technology.



Khady Ndoye

Management Information Systems-Accelerated Track

Ndoye is an international student from Senegal. She is super interested in data science and machine learning.



Hillary Neaves

Management Information Systems

Neaves is a junior from OKC, OK and she has always had an interest in Business. She has been curious from the start regarding the way that computers and technology work. She loves MIS and enjoys learning more about coding and programming.



Lam Nguyen

Finance - Minor: Management Information Systems

Nguyen is an international student from Vietnam. He has always been interested in business, particularly in finance and consulting. He hopes to learn more about programming throughout his professional career as he feels it is an important set of skill

Conceptual Design

The conceptual design phase is the first stage in a database design process which includes design interactions, experience, and strategies. The goal during this stage is to output a conceptual data model that describes the main data entities, relationships, and constraints of a given problem. The builder must discover the characteristic of the data element during this stage. The phase calls for information collection, developing and gathering end-user data views, observing the current system in place, and being in touch with the systems design group.

The Client Meeting

On October 12th at 9:45am, Team Mozart met with Frederic Daugherty, the liaison for Elysian Fly Company. Team members Sahrin Moytree, Sara Mullins, Khady Ndoye, Hillary Neaves, and Lam Nguyen were present for the 15-minute discussion. The interview resulted in the team accumulating crucial information regarding the structure of the desired database. Mr. Daugherty stated his company demands a database that is fully functional in modelling their entire business. Their key demands revolve around a well-managed inventory, accessible information regarding top vendors and employees, automatic generation of customer information and orders, better product control regarding their status and discounts, and exceptional understanding of customer data and demands to make valuable decisions for the company.

- Meeting Time: October 12th - 9:45am
- Location: Michael F. Price College of Business – Room 3107
- Interviewers: Sahrin Moytree, Sara Mullins, Khady Ndoye, Hillary Neaves, Lam Nguyen
- Interviewee: Frederic Daugherty

Q&A During the Meeting & Information We Learned:

1. What testing do products have to go through in order to be carried?

Yes, there must be an attribute which measures the passing capacity of a product.

2. Is there a large demand for the “bundled” package of flies?

Yes, there is a huge demand for bundled packages. The company asks the database to model bundled packages separately.

4. Do we include the mailing list in the ERD?

Yes, all new and current customers join our mailing lists. This is how the company keeps track of our customers and uses it as a form of publicity. The database needs to model all customers regardless of how many products they have purchased.

5. Do you want an outsource entity?

Yes, Elysian Fly would like to track the outsourced for 3rd party packages.

6. Are all tourist guides permanent employees or are they a different entity?

Yes, all tourist guides are considered employees of the company. Seasonal or not.

7. Are local families who provide hand tied flies considered vendors?

Yes, any merchandise obtained from outside of the company will be considered a vendor's product.

8. Do certain employee/ guide servers have access to certain types of information or all information available?

Employees/guides only have access to information pertaining to their job.

The section below provides crucial information regarding Entity Relationship Diagrams and their functionality. The reader will explore what an ERD is and how it is useful to companies all over the globe. The ERD for Elysian Fly Company has been published below. After careful accumulation of required entities and a thorough inspection of relationships among them, Team Mozart found several assumptions regarding the database. The clarification of the following assumptions will strongly affect the functionality of the database constructed for the company.

Significant Assumptions

- 1. What is the ChannelTypeID?** The ChannelTypeID is listed to include whether the order was an online order, a phone order, or an in-person order and has a relationship with SalesOrder.
- 2. Are all kinds of Employees accounted for?** Yes! All current, former, and future employees of the company are accounted for.
- 3. Can an employee have more than one sale?** Yes, an employee can have more than one sale. This is shown in the "zero-to-many" cardinality between Employee and SalesOrder.
- 4. Can a certain pattern or design be used on more than one fly? Or can it only be used once?** A pattern or design can be used multiple times, not just once. This is shown in the reference table of Pattern off of Fly. A pattern or design can be used multiple times, but each fly can only have one pattern or design. This assumption is the same for size and colour.

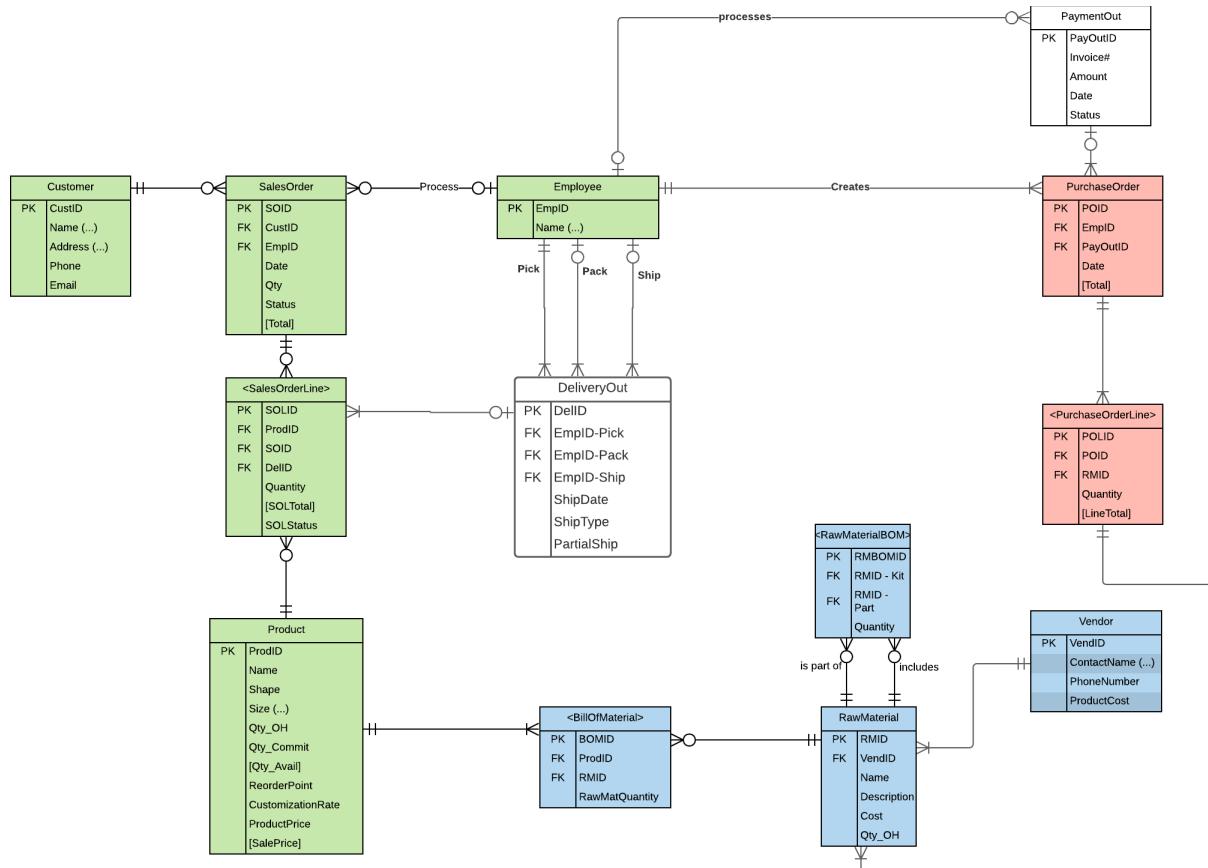
What is an ERD? Why is it necessary?

An Entity-Relationship diagram is a type of a structural diagram used in database design. The diagram calls for different symbols and connectors that visualize two important information: major entities within the project, and the inter-relationships among these entities. The data modeling technique is so in demand because it graphically illustrates an information system's entities and their relationships, thus, succeeding in effectively helping companies navigate the landscape of their business. The overview holistically, modelled by the entity-relationship diagram will help companies build the applications and resources, and communicate tasks needed to support their business.

Business Cycles Used

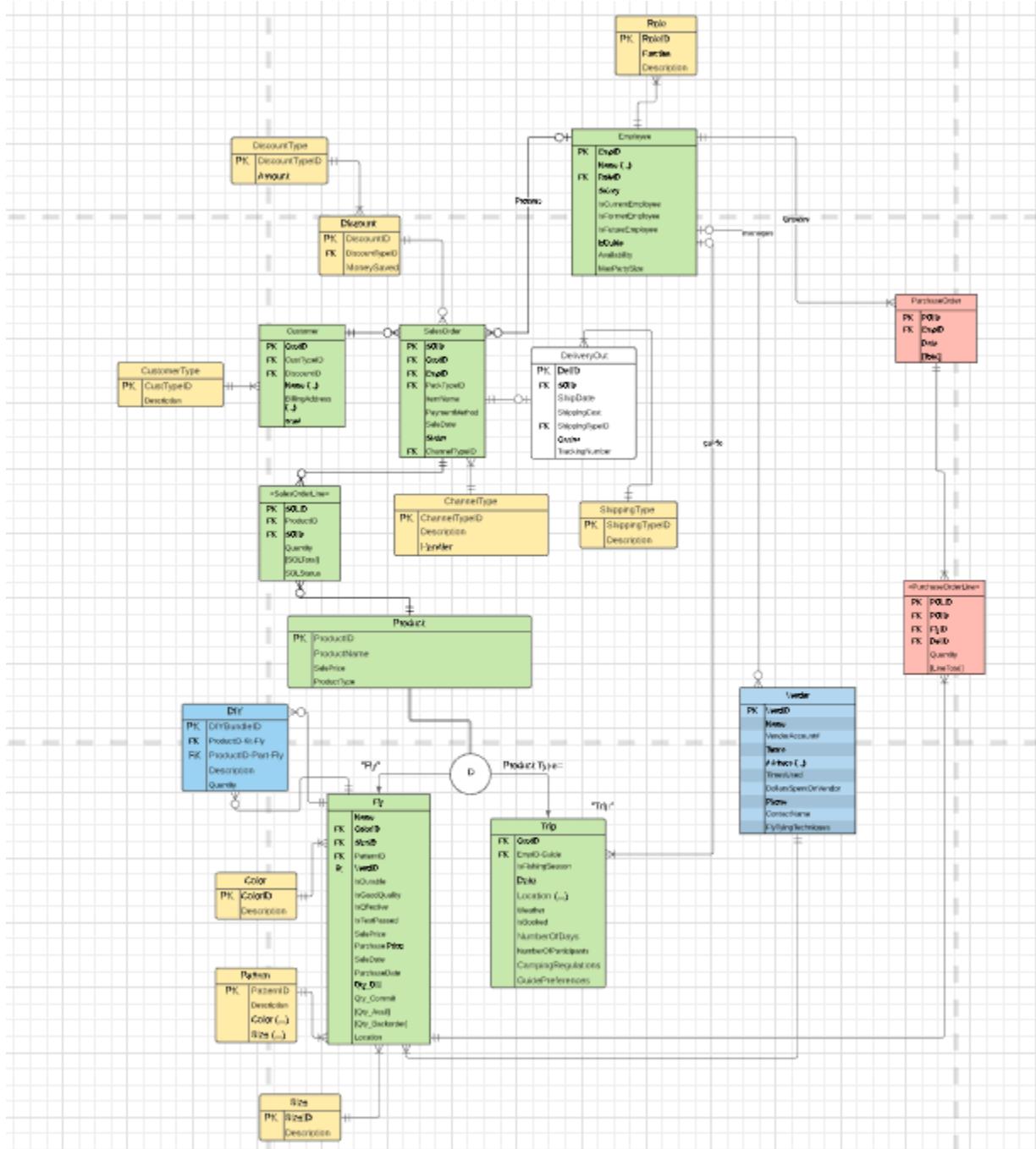
Three business cycles are used in the Entity Relationship diagram for Elysian Fly company. First and foremost, the **Production Cycle**, which is used since raw materials are being purchased from vendors to curate products for the company's inventory. However, only the unary many to many relationships is used from the production process. This is because there are many occurrences where two participants in a relationship are the same entity. The production process itself is not included. Second, the **Revenue Cycle** is utilized because the company is selling products. For example, the business of flies, trips, and bundles are all part of the revenue cycle. Lastly, the **Expenditure Cycle** is provided due to shipping, purchase order, and purchase of raw materials from vendors. The company is spending on necessary products.

Original Integrated Business Cycle

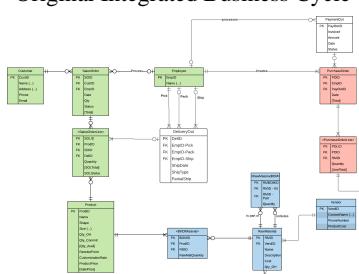
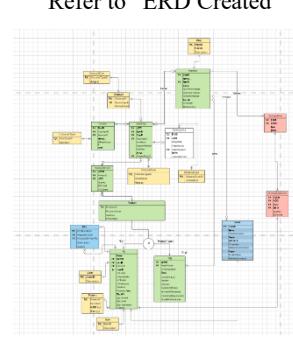
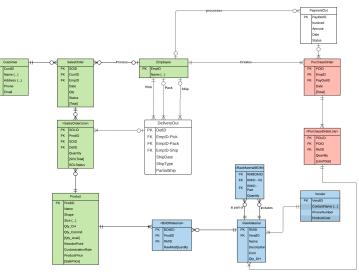
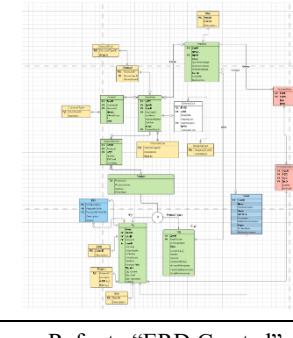
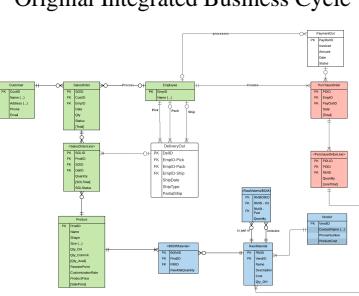
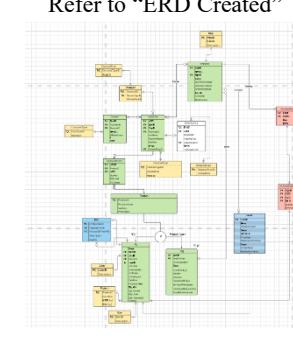
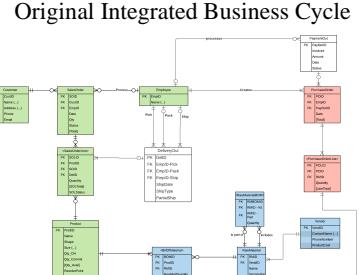
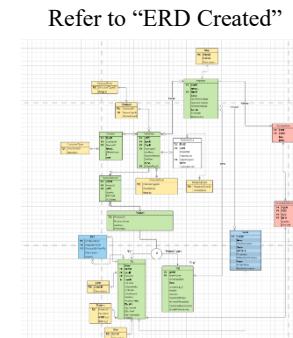


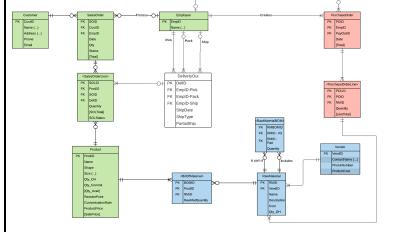
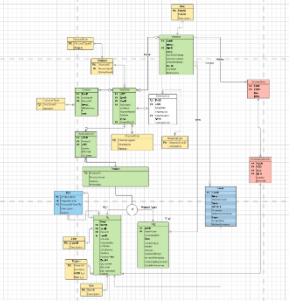
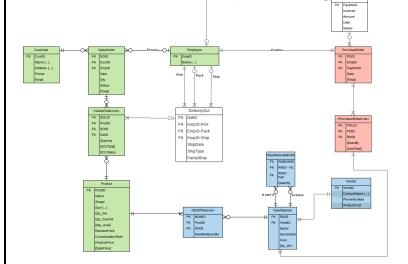
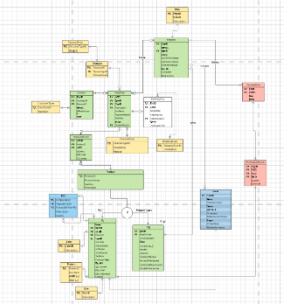
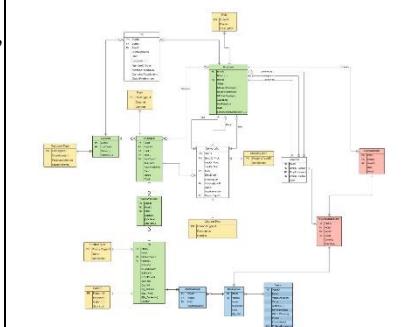
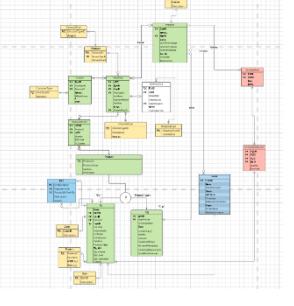
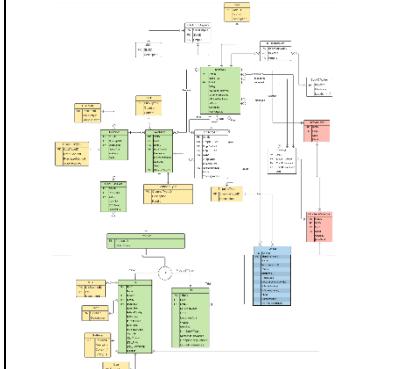
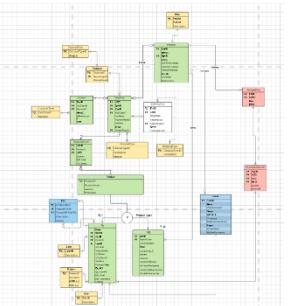
ERD Created

Elysian Fly Company



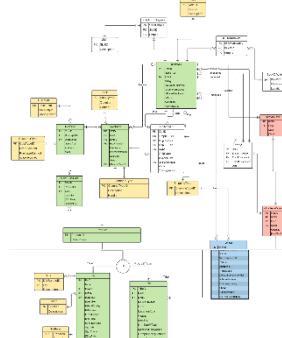
Changed made to generic ERDs

| Change # | Original ERD | Updated ERD |
|---|--|---|
| Entities were removed from original Integrated business cycles due to the information not pertaining to resources need. | Original Integrated Business Cycle  | Refer to "ERD Created"  |
| Addition of a few entities, attributes, and reference tables to the integrated business cycles. | Original Integrated Business Cycle  | Refer to "ERD Created"  |
| Generated list of assumptions for the original integrated business cycle and its functionality. We wanted to make sure that it was clear what each attribute or relationship meant. | Original Integrated Business Cycle  | Refer to "ERD Created"  |
| When adding reference tables, we had to add/change the cardinality of each relationship. We had to make sure that each entity was connected to its reference tables with the correct cardinality. | Original Integrated Business Cycle  | Refer to "ERD Created"  |

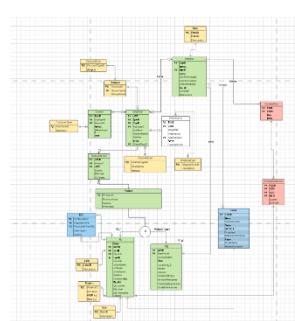
| | | |
|--|--|---|
| <p>Color coding the tables was another change that we made to our ERD. When adding tables, we had to color code them based on what cycle they were in or what cycle they connected to.</p> | <p>Original Integrated Business Cycle</p>  | <p>Refer to “ERD Created”</p>  |
| <p>Another change we made was changing the name of entities or reference tables to make it even more clear what we were listing. We thought that labeling the entities with names that pertained to the project, instead of just general names, would make it easier for the client to read.</p> | <p>Original Integrated Business Cycle</p>  | <p>Refer to “ERD Created”</p>  |
| <p>A major change that we made was adding sub-type/super-types to our ERD. In our original ERD, we created multiple reference tables instead of sub-type/super-types, but changed that in our updated one.</p> | <p>Milestone 1 ERD</p>  | <p>Refer to “ERD Created”</p>  |
| <p>A new change that we made was adding email and billing address in the Customer entity.</p> | <p>Milestone 2 ERD</p>  | <p>Refer to “ERD Created”</p>  |

Another change that we made was moving the relationship Discount had with Customer to Sales Order entity.

Milestone 2 ERD

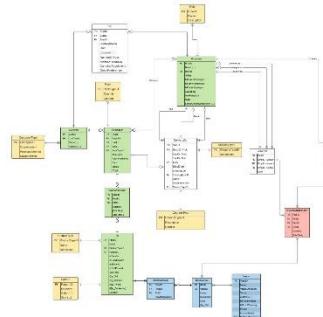


Refer to "ERD Created"

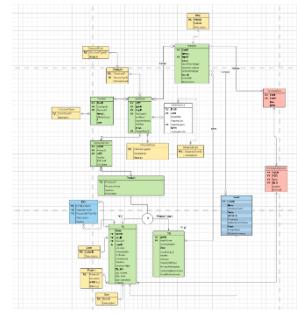


Another change that we made was fixing the cardinality on some of our entities. While learning more about our Company, we were able to make more informed decisions on the cardinality relationships.

Milestone 1 ERD



Refer to "ERD Created"



Logical Design

The logical database design is the simple process of deciding how to arrange the attributes of the entities in each business environments into database structures. The goal of logical database design is to create well-structured relational databases by converting entity-relationship diagrams into relational tables. The tables will manage to store data regarding a company's entities in a non-redundant manner, along with placing foreign keys so that all relationship among the entities will be supported.

Normalization

Normalization is the process of reorganizing data within a database to reduce data redundancy. This process is crucial because it allows databases to take up as little disk space as possible, therefore, creating increased performance. The first goal during data normalization is to detect and eliminate all duplicate data. This is done by logically grouping data redundancies together. Once all anomalies and unstructured data is processed into a structured form, normalization improves the usability of a data set. As redundancies are merged, data can be visualized more easily, insights can be extracted more efficiently, information can be updated more quickly, and the presence of errors much less. This process is important to the Elysian Fly Company because it will help Mozart do the implementation for the database. It is also needed to make SQL queries.

Normalized Relations

TState (StateID, StateName)

TZipCode (ZipCodeID, ZCity, ZStateID)

Foreign key ZStateID references TState
Not Null
On Delete Restrict

TCustomerType(CustTypeID, CTDescription)

TDiscountType(DiscountTypeID, DTDescription)

TDiscount(DiscountID, DDiscountTypeID, DMoneySaved)

Foreign key DDiscountTypeID references TDiscountType
Not Null
On delete restrict

TCustomer (CustID, CustTypeID, CustZipCodeID, CustFirstName, CustLastName, Custemail)

Foreign key CustTypeID references TCustomerType
Not Null
On Delete Restrict
Foreign key CustZipCodeID references TZipCode
Not Null
On Delete Restrict

TRole (RoleID, RFunction, RDescription)

TEmployee (EmpID, EmpRoleID, EmpFirstName, EmpLastName, EmpSalary, IsCurrentEmployee, IsFormerEmployee, IsFutureEmployee, IsGuide, EmpAvailability, EmpMayPartySize)

Foreign key EmpRoleID references TRole

Not Null

On Delete Restrict

TChannelType(ChannelTypeID, ChanDescription, ChanHandler)

TSalesOrder (SOID, SOCustID, SOEmpID, SOPackTypeID, SOChannelTypeID, SODiscountID, SOItemName, SOPaymentMethod, SOSaleDate, SOStatus)

Foreign key SOCustID references TCustomer

Not Null

On Delete Restrict

Foreign key SOEmpID references TEmployee

Null Allowed

On Delete Cascade

Foreign key SOPackTypeID references TPack

Not Null

On Delete Restrict

Foreign key SOChannelTypeID references TChannelType

Not Null

On Delete Restrict

Foreign key SODiscountID references TDdiscount

Not Null

On Delete Restrict

TProduct (ProductID, ProdName, SalePrice, ProductType)

TSalesOrderLine (SOLID, SOLProductID, SOLSOID, SOLQuantity, SOLStatus)

Foreign Key SOLProductID references TProduct

Not Null

On Delete Restrict

Foreign Key SOLSOID references TSalesOrder

Not Null

On Delete Restrict

TColor(ColorID, ColorDescription)

TPattern(PatternID, PatDescription, PatColor, PatSize)

TSize(SizeID, SizeDescription)

TDIY(DIYBundleID, DIYFlyProductID-Kit, DIYFlyProductID-Part, DIYDescription, DIYQuantity)

Foreign Key DIYFlyProductID-Kit references TFly

Not Null

On Delete Restrict

Foreign Key DIYFlyProductID-Part references TFly

Not Null

On Delete Restrict

TFly(FlyProductID, Flyname, FlyColorID, FlySizeID, FlyPatternID, FlyisDurable, FlyisGoodQuality, FlyisEffective, FlyisTestPassed, FlySalesPrice, FlyPurchaseCost, FlyQty_OH, FlyQty_Commit, FlyQty_Avail, FlyQty_Backorder, FlyLocation, PurchaseDate, SaleDate)

Foreign Key FlyColorID references TColor

Not Null

On Delete Restrict

Foreign Key FlySizeID references TSize

Not Null

On Delete Restrict

Foreign Key FlyPatternID references TPattern

Not Null

On Delete Restrict

TTrip(TripProductID, TripCustID, TripGuideEmpID, TripisFishingSeason, TripDate, TripZipCodeID, TripWeather, TripisBooked, TripNumofDays, TripNumofParticipants, TripRegulations, TripGuidePreferences) key

Foreign Key TripCustID references TCustomer

Not Null

On Delete Restrict

Foreign Key TripGuideEmpID references TEmployee

Not Null

On Delete Restrict

Foreign Key TripZipCodeID references TZipCode

Not Null

On Delete Restrict

TShippingType(ShippingTypeID, SPTDescription)

TDeliveryOut(DellID, DOSOID, DOShipDate, DOShippingCost, DOShippingTypeID, DOCarrier, DOTrackingNumber)

Foreign Key DOSOID references TSalesOrder

Not Null

On Delete Restrict

Foreign Key DOShippingTypeID references TShippingType

Not Null

On Delete Restrict

TPurchaseOrder(POID, POEmpID, PODate, POTotal)

Foreign Key POEmpID references TEmployee

Not Null

On Delete Restrict

TPurchaseOrderLine(POLID, POLPOID, POLFlyID, POLDellID, POLQuantity, POLLineTotal)

Foreign Key POLPOID references TPurchaseOrder

Not Null

On Delete Restrict

Foreign Key POLFlyID references TFly

Not Null

On Delete Restrict

Foreign Key POLDellID references TDeliveryIn

Not Null

On Delete Restrict

**TVendor(VendID, VName, VVendorAccount#, VTerms, VZipCodeID, VTimesUsed,
VDollarsSpentOnVendor, VPhone, VContactName, VFlyingTechniques)**

Foreign Key VZipCodeID references TZipCode

Not Null

On Delete Restrict

Differences between ERD and Normalized Relations

An entity relationship diagram defines the relationship between entities and their attributes. Normalization occurs when a database maintains a set of separate, related files (tables), but combines data elements from the files for queries and reports when required. In short, an ERD is an abstract concept of a database, it speaks in entities and attributes. A normalization relational model defines formats and relations in a way a database could understand, a data model. The three key differences between ERD and normalized relations are atomicity, no data duplication issues, and well-structured data. The differences made between the Entity Relationship Diagram and normalization relations for Elysian Fly company are minor yet provides a much more efficient database output. First and foremost, 'Location' was changed by creating TZipCode and TState to avoid data redundancy. Second, 'Name' is composite in the Elysian Fly Company ERD for the entities of 'EMPLOYEE' and 'CUSTOMER', therefore, it was split into 'FirstName' and 'LastName' in the logical design to avoid partial functional dependency. Lastly, the subtypes, Fly and Trip, were given primary keys to specify them. It is beneficial to have normalized relations because it provides a more effective customer profiles, optimized internal resources, reduced response time and offers additional guarantee.

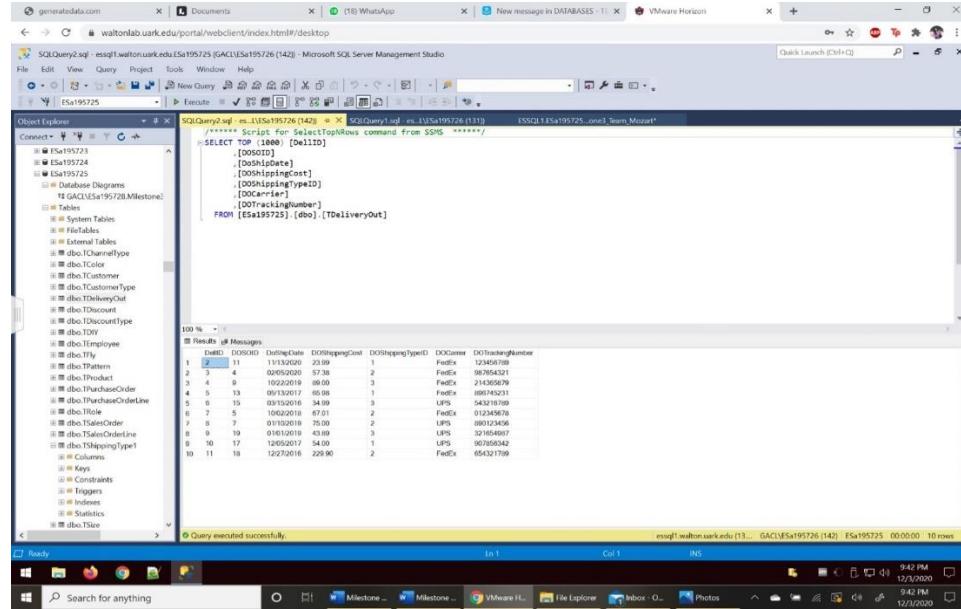
Referential Integrity

Referential integrity refers to the accuracy and consistency of a data within a relationship. The rule requires that if a foreign key value is used, it must reference a valid and existing primary key in the parent table (primary key). Referential integrity prevents users from adding data to a related table if there is no associated data in the primary table, changing values in a primary table that results in orphaned data in a related table, and deleting data from a primary table if there are no related records. The lack of referential integrity in a database can lead to incomplete data being returned with no notification of error. This could lead to data being lost in the database because they are never received in queries or reports. It is significant to the case of the Elysian Fly Company because the lack of referential integrity can cause problems such as customers not receiving orders they paid for. Two other integrity constraints are domain and entity. A domain constraint can be defined as the definition of a valid set of values for an attribute. The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain. Lastly, the entity integrity requires that each entity has a unique key. For example, if every row in a table represents relationships for a unique entity, the table should have one column or a set of columns that provides a unique identifier for the rows of the table. In simpler words, the entity integrity's focus is to define the parent key

Physical Design and Implementation

Below, you will find screenshots of the different tables we have created in our physical design.

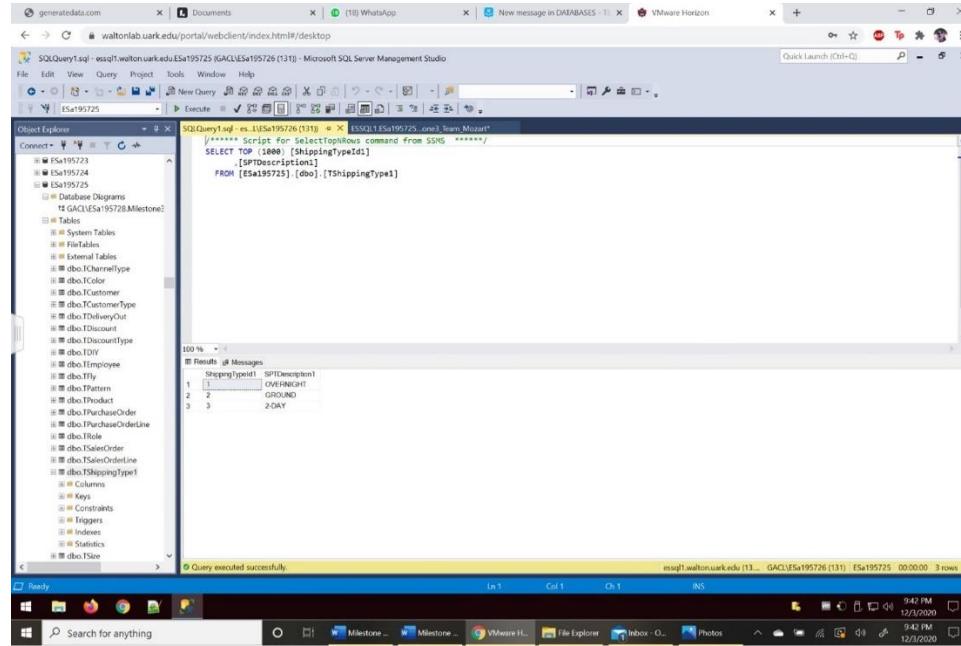
TDeliveryOut:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including tables like TDeliveryOut, TCustomer, and TShippingType. The central pane displays a query results grid for the TDeliveryOut table. The columns are DODate, DOSSN, DODingDate, DOIShippingCost, DOIShippingTypeID, DOCarrier, and DOTrackingNumber. The data shows 10 rows of shipping information, such as FedEx and UPS services.

| DODate | DOSSN | DODingDate | DOIShippingCost | DOIShippingTypeID | DOCarrier | DOTrackingNumber |
|--------|-------|------------|-----------------|-------------------|-----------|-------------------|
| 1 | 2 | 11 | 1/13/2009 | 23.99 | 1 | FedEx 123456789 |
| 2 | 3 | 4 | 1/13/2009 | 57.98 | 2 | FedEx 987654321 |
| 3 | 4 | 6 | 1/25/2010 | 66.99 | 3 | FedEx 21098765432 |
| 4 | 4 | 13 | 0/13/2012 | 66.98 | 1 | FedEx 0987654321 |
| 5 | 6 | 15 | 0/15/2016 | 34.99 | 3 | UPS 543210989 |
| 6 | 7 | 5 | 1/02/2018 | 67.01 | 2 | FedEx 012345678 |
| 7 | 7 | 2 | 1/02/2018 | 67.00 | 2 | UPS 32109876543 |
| 8 | 9 | 19 | 0/16/2019 | 43.99 | 3 | UPS 32109876543 |
| 9 | 10 | 17 | 1/26/2017 | 54.00 | 1 | UPS 987654321 |
| 10 | 11 | 18 | 1/27/2018 | 22.90 | 2 | FedEx 654321098 |

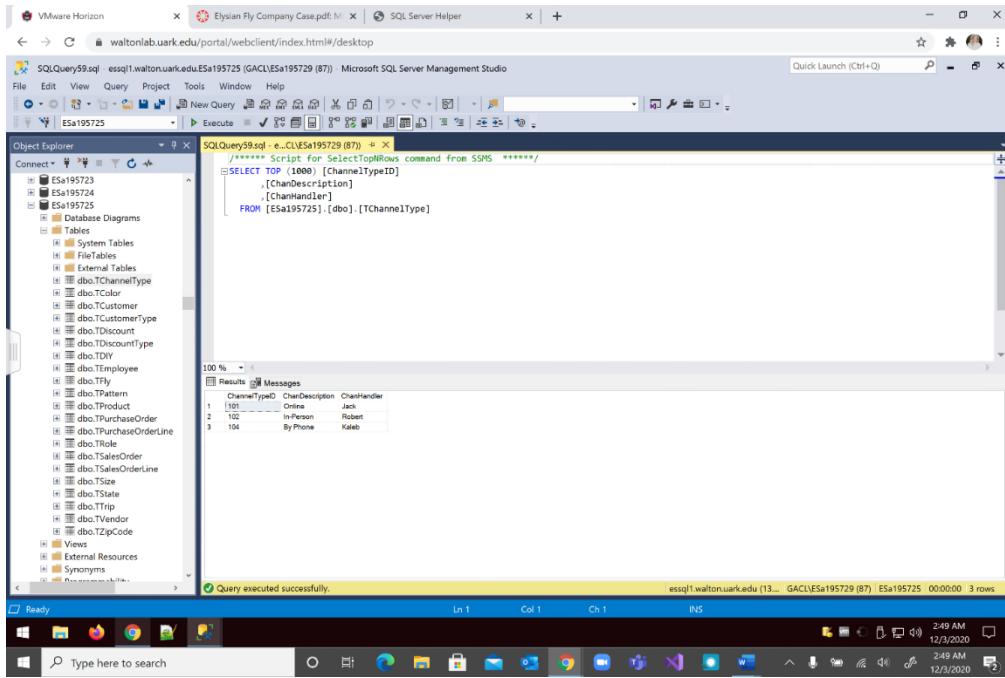
TShippingType:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left shows the database structure, including tables like TShippingType and TCustomer. The central pane displays a query results grid for the TShippingType table. The columns are ShippingTypeID and SPTDescription. The data shows three rows of shipping type descriptions, such as GROUND and 2-DAY.

| ShippingTypeID | SPTDescription |
|----------------|----------------|
| 1 | GROUND |
| 2 | 2-DAY |
| 3 | 3-DAY |

TChannelType:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like TChannelType, TColor, and TCustomer. The central pane displays the results of a query:

```

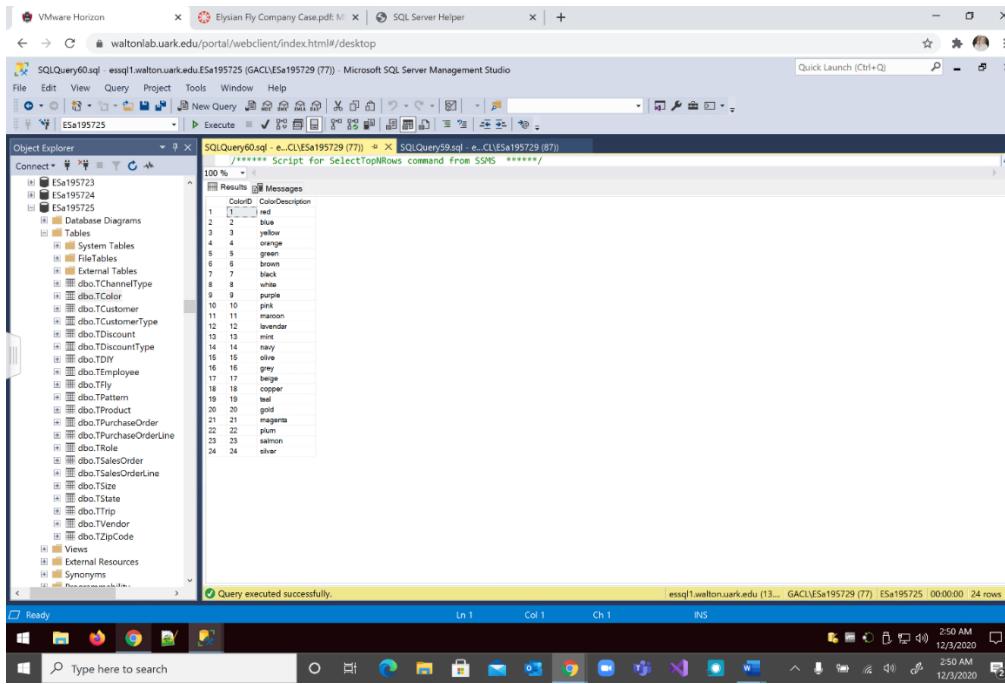
SELECT TOP (1000) [ChannelTypeID]
      ,[ChanDescription]
      ,[ChanHandler]
   FROM [ESa195725].[dbo].[TChannelType]
  
```

The results table shows three rows:

| ChannelTypeID | ChanDescription | ChanHandler |
|---------------|-----------------|-------------|
| 101 | Online | Jack |
| 102 | Print | Roger |
| 104 | ByPhone | Kaleb |

Below the results, a message indicates "Query executed successfully." The status bar at the bottom shows the connection details: esql1.walton.ark.edu (13... GACL\ESa195725 (87) ESa195725 00:00:00 3 rows".

TColor:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like TChannelType, TColor, and TCustomer. The central pane displays the results of a query:

```

SELECT TOP (1000) [ColorID]
      ,[ColorDescription]
   FROM [ESa195725].[dbo].[TColor]
  
```

The results table shows 24 rows of color names and their corresponding IDs:

| ColorID | ColorDescription |
|---------|------------------|
| 1 | red |
| 2 | blue |
| 3 | yellow |
| 4 | orange |
| 5 | green |
| 6 | brown |
| 7 | black |
| 8 | white |
| 9 | purple |
| 10 | pink |
| 11 | brown |
| 12 | lavender |
| 13 | mint |
| 14 | navy |
| 15 | olive |
| 16 | gray |
| 17 | beige |
| 18 | copper |
| 19 | teal |
| 20 | gold |
| 21 | magenta |
| 22 | plum |
| 23 | tan |
| 24 | silver |

Below the results, a message indicates "Query executed successfully." The status bar at the bottom shows the connection details: esql1.walton.ark.edu (13... GACL\ESa195725 (77) ESa195725 00:00:00 24 rows".

TCustomer:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like TCustomer, TCustomerType, and TDOrder. The Results pane on the right displays the output of a query on the TCustomer table, showing columns such as CustID, CustTypeID, CustCode, CustName, and CustLastname. The results show 100 rows of customer data.

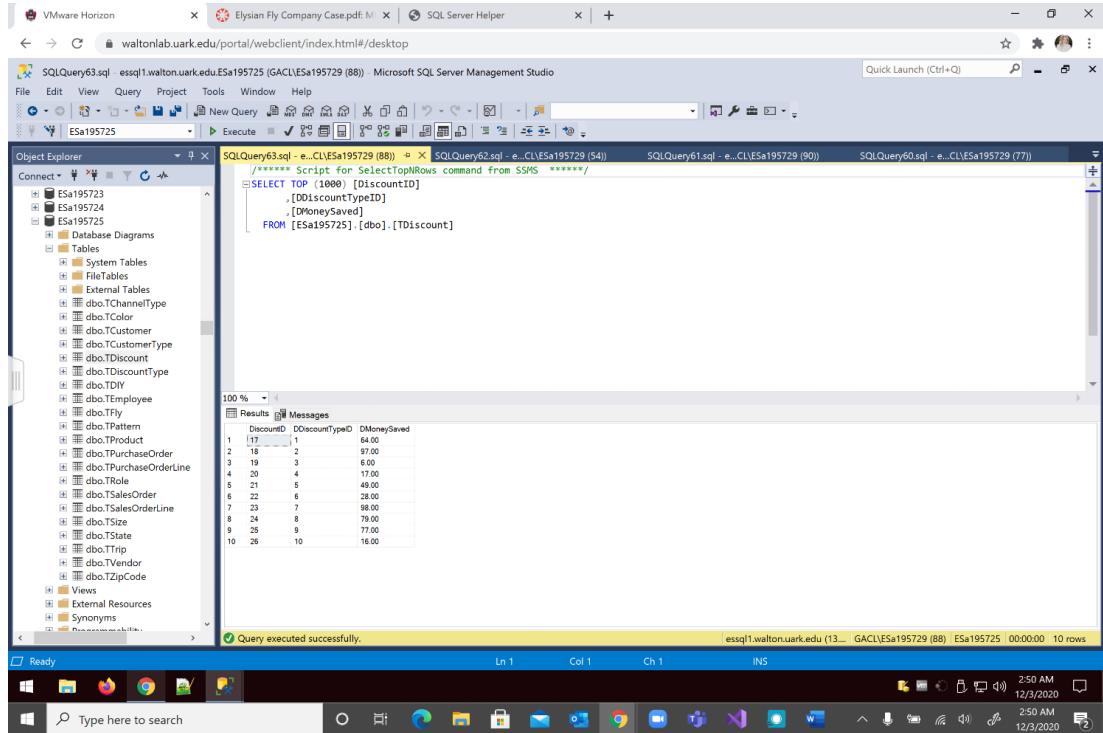
| CustID | CustTypeID | CustCode | CustName | CustLastname |
|--------|------------|----------|----------|--------------|
| 1 | 8 | 2 | Jack | Morton |
| 2 | 18 | 3 | Lam | Nguyen |
| 3 | 48 | 1 | Brad | Bradley |
| 4 | 49 | 3 | James | John |
| 5 | 50 | 2 | Peter | Parker |
| 6 | 51 | 1 | Clark | Kent |
| 7 | 52 | 2 | Steve | Wayne |
| 8 | 53 | 3 | Kendall | Woods |
| 9 | 54 | 1 | Courtesy | Hunter |
| 10 | 55 | 2 | David | Black |
| 11 | 262 | 9 | Jonah | Schneider |
| 12 | 263 | 3 | Ian | Trevor |
| 13 | 284 | 2 | Kathleen | Hartman |
| 14 | 285 | 2 | Ginger | Beyd |
| 15 | 286 | 1 | Calvin | Parker |
| 16 | 287 | 2 | Derek | Green |
| 17 | 288 | 3 | Quinn | Sanders |
| 18 | 289 | 1 | Jaden | Green |
| 19 | 290 | 1 | Wylie | Henderson |
| 20 | 291 | 1 | Lila | Rush |
| 21 | 292 | 1 | Jordan | Matthews |
| 22 | 293 | 2 | Sean | Townsend |
| 23 | 294 | 1 | Karen | Rosa |
| 24 | 295 | 1 | Isaac | Mueller |
| 25 | 296 | 2 | Colton | Fisher |
| 26 | 297 | 2 | Olivia | Hernandez |
| 27 | 298 | 3 | Feris | Cox |
| 28 | 299 | 1 | Murphy | Michael |
| 29 | 300 | 3 | Antony | Alderman |
| 30 | 301 | 2 | Margaret | Holder |
| 31 | 302 | 2 | Stacy | Heath |
| 32 | 303 | 2 | Venita | Goodwin |
| 33 | 304 | 1 | Jordan | James |
| 34 | 305 | 3 | Brenna | Brown |
| 35 | 306 | 2 | Cara | Mccormick |
| 36 | 307 | 3 | Deneius | Paul |
| 37 | 308 | 3 | | |

TCustomerType:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists the database structure, including tables like TCustomer, TCustomerType, and TDOrder. The Results pane on the right displays the output of a query on the TCustomerType table, showing columns such as CustTypeID and CTDescription. The results show 3 rows of customer type data.

| CustTypeID | CTDescription |
|------------|-------------------|
| 1 | One-Off Tourist |
| 2 | Discount Customer |
| 3 | Local |

TDiscount:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases Esa195723, Esa195724, and Esa195725. The Results pane displays the output of a query:

```

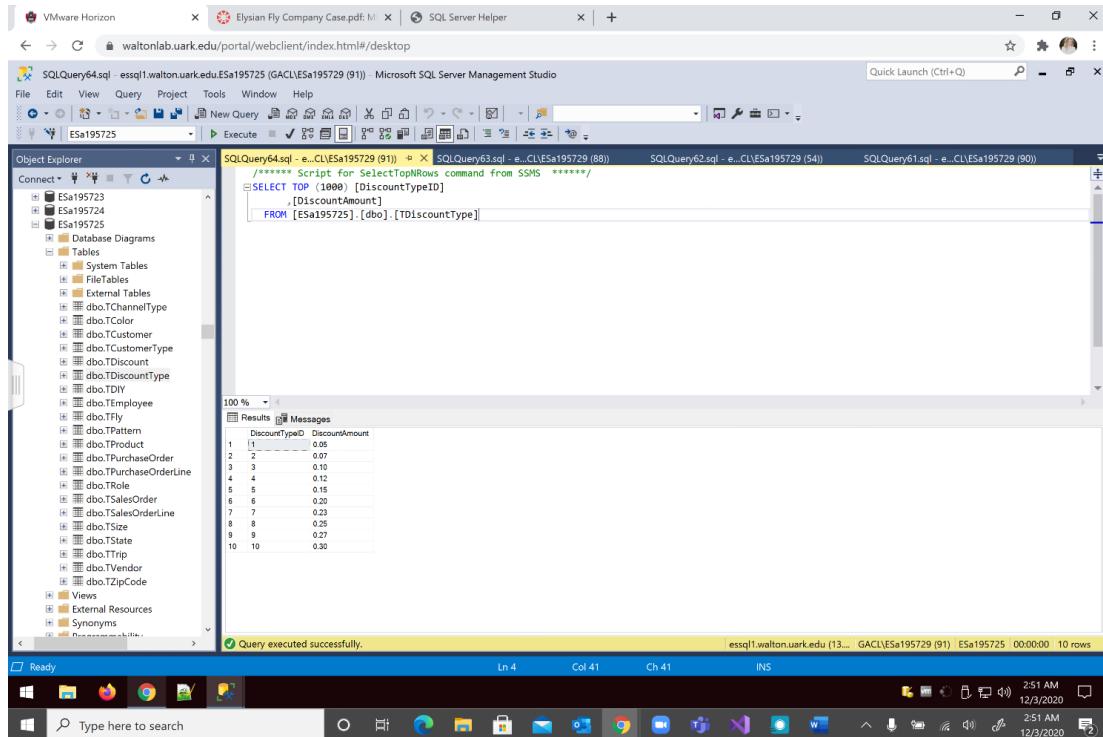
SELECT TOP (1000) [DiscountID]
      ,[DiscountTypeID]
      ,[DMoneySaved]
   FROM [Esa195725].[dbo].[TDiscount]
  
```

The results table has columns DiscountID, DiscountTypeID, and DMoneySaved, with 10 rows of data:

| DiscountID | DiscountTypeID | DMoneySaved |
|------------|----------------|-------------|
| 1 | 17 | 64.00 |
| 2 | 18 | 97.00 |
| 3 | 19 | 6.00 |
| 4 | 20 | 17.00 |
| 5 | 21 | 40.00 |
| 6 | 22 | 28.00 |
| 7 | 23 | 98.00 |
| 8 | 24 | 79.00 |
| 9 | 25 | 77.00 |
| 10 | 26 | 16.00 |

At the bottom, it says "Query executed successfully."

TDiscountType:



The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists databases Esa195723, Esa195724, and Esa195725. The Results pane displays the output of a query:

```

SELECT TOP (1000) [DiscountTypeID]
      ,[DiscountAmount]
   FROM [Esa195725].[dbo].[TDiscounType]
  
```

The results table has columns DiscountTypeID and DiscountAmount, with 10 rows of data:

| DiscountTypeID | DiscountAmount |
|----------------|----------------|
| 1 | 0.05 |
| 2 | 0.07 |
| 3 | 0.10 |
| 4 | 0.12 |
| 5 | 0.15 |
| 6 | 0.20 |
| 7 | 0.23 |
| 8 | 0.25 |
| 9 | 0.27 |
| 10 | 0.30 |

At the bottom, it says "Query executed successfully."

TDIY:

| DITBndlID | DITProductID_Prt | DITProductID_KD | DITDescription | DITQuantity |
|-----------|------------------|-----------------|--------------------------|-------------|
| 1 | 2 | 2 | Buy One-Get One Combo | 4 |
| 2 | 2 | 4 | Suitable for Beginners | 7 |
| 3 | 4 | 3 | Assemble at Ease | 13 |
| 4 | 5 | 4 | For Professionals | 6 |
| 5 | 6 | 5 | Family Package | 2 |
| 6 | 6 | 2 | Time Saving | 15 |
| 7 | 9 | 7 | Experience Required | 20 |
| 8 | 11 | 8 | For Children | 12 |
| 9 | 12 | 9 | For Adults | 9 |
| 10 | 13 | 10 | Stocks Buffer | 5 |
| 11 | 18 | 21 | Supplies Needed | 18 |
| 12 | 19 | 12 | Wire Required | 27 |
| 13 | 20 | 13 | Quick and Simple | 2 |
| 14 | 21 | 14 | Precise and Perfect | 25 |
| 15 | 22 | 15 | Activity for Friends | 10 |
| 16 | 23 | 16 | Generation to Generation | 30 |
| 17 | 24 | 17 | Heavy Duty | 56 |
| 18 | 25 | 18 | Plies for Dummies | 32 |
| 19 | 26 | 19 | For the Old | 16 |
| 20 | 27 | 20 | Beginner Luck | 8 |

Query executed successfully.

TEmployee:

| EmpID | EmpFirstName | EmpLastName | EmpSalary | IsCurrentEmployee | IsFormerEmployee | IsFutureEmployee | IsGuide | EmpAvailability | EmpMayPartySize | EmpRoleID |
|-------|--------------|-------------|-----------|-------------------|------------------|------------------|---------|-----------------|-----------------|-----------|
| 1 | Mark | Miles | 50000 | YES | NO | NO | NO | 0 | 2 | |
| 2 | Robert | Miles | 40000 | YES | NO | NO | YES | 0 | 4 | |
| 3 | Tomma | Banbara | 50000 | YES | NO | NO | YES | 0 | 5 | |
| 4 | Sidi | Ganane | 45000 | YES | NO | NO | YES | 0 | 6 | |
| 5 | Laure | Maure | 47000 | NO | NO | YES | YES | 10 | 9 | |
| 6 | Lea | Sanchaz | 50000 | YES | NO | NO | YES | 0 | 7 | |
| 7 | Sam | Ly | 62000 | YES | NO | NO | YES | 0 | 8 | |
| 8 | Ryan | Smith | 42000 | YES | NO | YES | NO | 10 | 2 | |
| 9 | Tony | Lan | 40000 | YES | NO | NO | YES | 20 | 9 | |
| 10 | Jack | Ruark | 50000 | NO | NO | YES | YES | 10 | 5 | |
| 11 | Sara | Lee | 21000 | YES | YES | NO | YES | 7 | 8 | |
| 12 | Seen | Thompson | 29000 | YES | NO | NO | NO | 8 | 12 | |
| 13 | Kate | Wright | 39000 | NO | YES | NO | NO | 10 | 1 | |
| 14 | Lucy | Seize | 13000 | NO | NO | YES | NO | 11 | 19 | |
| 15 | Mike | Faend | 50000 | YES | NO | NO | YES | 2 | 9 | |
| 16 | Brayden | West | 7000 | NO | NO | YES | YES | 18 | 11 | |
| 17 | Bennett | Price | 8200 | YES | NO | NO | YES | 9 | 17 | |
| 18 | Susan | Nice | 55000 | NO | NO | YES | NO | 16 | 78 | |
| 19 | Landon | Loery | 67000 | YES | NO | NO | YES | 89 | 12 | |
| 20 | Logan | Peter | 54000 | NO | YES | NO | YES | 13 | 22 | |

Query executed successfully.

TFly:

| FlyProductID | ProductID | FlyName | FlyColorID | FlySizeID | FlyPattern | FlyDurable | FlyGoodQuality | FlyEffective | FlyTestPassed | FlySalesPrice | FlyPurchaseCost | FlyQty_OH | FlyQty__Committ | FlyQty_Avail | FlyQty_BackOrder | FlyLoc | FlyLastUpdate |
|--------------|-----------|---------|------------|-----------|------------|------------|----------------|--------------|---------------|---------------|-----------------|-----------|-----------------|--------------|------------------|---------|-------------------------|
| 1 | 1 | FlyA | 1 | 5 | 1 | Yes | Yes | Yes | Yes | 20 | 25 | 60 | 10 | 40 | 30 | Storage | 2020-12-01 12:00:00.000 |
| 2 | 2 | FlyB | 2 | 3 | 2 | Yes | Yes | Yes | Yes | 50 | 40 | 77 | 15 | 35 | 31 | Storage | 2020-12-01 12:00:00.000 |
| 3 | 3 | FlyC | 3 | 11 | 3 | Yes | Yes | Yes | Yes | 40 | 30 | 100 | 70 | 30 | 26 | Storage | 2020-12-01 12:00:00.000 |
| 4 | 4 | FlyD | 4 | 12 | 4 | Yes | Yes | Yes | Yes | 60 | 55 | 15 | 30 | 20 | 16 | Storage | 2020-12-01 12:00:00.000 |
| 5 | 5 | FlyE | 5 | 13 | 5 | No | No | No | No | 25 | 20 | 5 | 0 | 5 | 0 | Storage | 2020-12-01 12:00:00.000 |
| 6 | 6 | FlyF | 6 | 14 | 101 | Yes | No | No | No | 70 | 27 | 7 | 1 | 38 | 38 | Storage | 2020-12-01 12:00:00.000 |
| 7 | 7 | FlyG | 7 | 15 | 136 | Yes | Yes | No | No | 32 | 32 | 35 | 15 | 99 | 37 | Storage | 2020-12-01 12:00:00.000 |
| 8 | 8 | FlyH | 8 | 16 | 10 | No | No | Yes | Yes | 16 | 26 | 120 | 20 | 100 | 89 | Storage | 2020-12-01 12:00:00.000 |
| 9 | 9 | FlyI | 9 | 17 | 6 | No | No | Yes | Yes | 89 | 39 | 20 | 77 | 22 | 99 | Storage | 2020-12-01 12:00:00.000 |
| 10 | 10 | FlyJ | 10 | 18 | 18 | No | No | Yes | Yes | 90 | 12 | 85 | 88 | 36 | 90 | Storage | 2020-12-01 12:00:00.000 |
| 11 | 11 | FlyK | 11 | 19 | 112 | Yes | No | Yes | Yes | 12 | 22 | 28 | 99 | 38 | 13 | Storage | 2020-12-01 12:00:00.000 |
| 12 | 12 | FlyL | 12 | 20 | 118 | Yes | No | No | No | 24 | 45 | 60 | 12 | 89 | 15 | Storage | 2020-12-01 12:00:00.000 |
| 13 | 13 | FlyM | 13 | 21 | 131 | Yes | No | Yes | Yes | 28 | 67 | 70 | 34 | 31 | 60 | Storage | 2020-12-01 12:00:00.000 |
| 14 | 14 | FlyN | 14 | 22 | 143 | No | Yes | No | Yes | 46 | 89 | 42 | 54 | 65 | 22 | Storage | 2020-12-01 12:00:00.000 |
| 15 | 15 | FlyO | 15 | 23 | 102 | Yes | Yes | Yes | Yes | 29 | 33 | 89 | 67 | 55 | 66 | Storage | 2020-12-01 12:00:00.000 |
| 16 | 16 | FlyP | 16 | 24 | 111 | No | No | Yes | Yes | 11 | 26 | 90 | 55 | 37 | 40 | Storage | 2020-12-01 12:00:00.000 |
| 17 | 17 | FlyQ | 17 | 18 | 162 | No | Yes | Yes | No | 17 | 54 | 22 | 78 | 28 | 100 | Storage | 2020-12-01 12:00:00.000 |
| 18 | 18 | FlyR | 18 | 2 | 181 | No | No | Yes | Yes | 0 | 42 | 57 | 78 | 21 | 5 | Storage | 2020-12-01 12:00:00.000 |
| 19 | 19 | FlyS | 19 | 3 | 176 | No | Yes | No | No | 55 | 73 | 99 | 44 | 96 | 77 | Storage | 2020-12-01 12:00:00.000 |
| 20 | 20 | FlyT | 20 | 11 | 135 | Yes | Yes | No | No | 68 | 29 | 82 | 90 | 44 | 34 | Storage | 2020-12-01 12:00:00.000 |

TPattern:

| PatternID | PatDescription | PatColor | PatSize |
|-----------|-------------------------|-------------------|---------|
| 1 | Moon | Yellow | 12 |
| 2 | Moon | White | 13 |
| 3 | Rainbow | Red Purple | 28 |
| 4 | Adams parachute | Pink | 12 |
| 5 | Adams parachute Special | Yellow | 17 |
| 6 | Adams parachute Special | Gold | 9 |
| 7 | DustTheWind | Brown | 18 |
| 8 | Abc | Green | 20 |
| 9 | James Parachute | Green | 19 |
| 10 | King o' Bay | Gold | 18 |
| 11 | half drop | teal | 26 |
| 12 | half drop | lavender | 11 |
| 13 | zebra | blue | 9 |
| 14 | plaid | salmon | 22 |
| 15 | purple stripe | teal | 10 |
| 16 | pin dot | lavender | 7 |
| 17 | 106 | half drop | 28 |
| 18 | 107 | pin dot | magenta |
| 19 | 108 | zebra | 8 |
| 20 | 109 | brick | 26 |
| 21 | 110 | gingham | 10 |
| 22 | 111 | sun | olive |
| 23 | 112 | pink dots | 6 |
| 24 | 113 | dots | 24 |
| 25 | 114 | nursery | yellow |
| 26 | 115 | rainbow | black |
| 27 | 116 | horizontal stripe | yellow |
| 28 | 117 | orange | 22 |
| 29 | 118 | quarter foil | brown |
| 30 | 119 | oriental | white |
| 31 | 120 | chevron | gold |
| 32 | 121 | Adams Parachute | yellow |
| 33 | 122 | bird's eye | salmon |
| 34 | 123 | pink stripe | beige |
| 35 | 124 | pinstripe | navy |
| 36 | 125 | yellow chevron | mint |
| 37 | 126 | pin stripe | tan |
| 38 | 127 | orange | 17 |

TProduct:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including Es195723, Es195724, and Es195725. The current database is Es195725. The Results pane displays the output of a query on the TProduct table, which contains 100 rows of product information. The columns shown are ProductID, ProducName, SalePrice, and ProductType. The data includes various products like FlyA, FlyB, FlyC, FlyD, FlyE, FlyF, FlyG, FlyH, FlyI, FlyJ, FlyK, FlyL, FlyM, FlyN, FlyO, FlyP, FlyQ, FlyR, FlyS, FlyT, FlyU, FlyV, FlyW, FlyX, FlyY, FlyZ, TripA, TripB, TripC, TripD, TripE, TripF, TripG, TripH, TripI, TripJ, TripK, TripL, TripM, TripN, TripO, TripP, TripQ, TripR, TripS, TripT, TripU, TripV, TripW, TripX, TripY, TripZ, and TripAA.

| ProductID | ProducName | SalePrice | ProductType |
|-----------|------------|-----------|-------------|
| 1 | FlyA | 30 | Fly |
| 2 | FlyB | 30 | Fly |
| 3 | FlyC | 40 | Fly |
| 4 | FlyD | 25 | Fly |
| 5 | FlyE | 35 | Fly |
| 6 | FlyF | 60 | Trip |
| 7 | TripG | 50 | Trip |
| 8 | TripH | 40 | Trip |
| 9 | TripI | 60 | Trip |
| 10 | TripJ | 50 | Trip |
| 11 | TripK | 30 | Trip |
| 12 | TripL | 32 | Trip |
| 13 | FlyCC | 21 | Fly |
| 14 | FlyDD | 39 | Fly |
| 15 | FlyEE | 48 | Fly |
| 16 | TripGG | 100 | Trip |
| 17 | TripHH | 28 | Trip |
| 18 | TripCC | 31 | Trip |
| 19 | TripDD | 80 | Trip |
| 20 | TripEE | 100 | Trip |
| 21 | FlyAB | 23 | Fly |
| 22 | FlyAC | 26 | Fly |
| 23 | FlyAD | 18 | Fly |
| 24 | FlyAE | 72 | Fly |
| 25 | FlyAF | 25 | Fly |
| 26 | FlyAG | 24 | Fly |
| 27 | FlyAH | 67 | Fly |
| 28 | FlyAI | 74 | Fly |
| 29 | FlyAJ | 53 | Fly |
| 30 | FlyAK | 33 | Fly |
| 31 | FlyAL | 56 | Fly |
| 32 | FlyAM | 99 | Fly |
| 33 | FlyAH | 78 | Fly |
| 34 | FlyAO | 72 | Fly |
| 35 | FlyAP | 45 | Fly |
| 36 | FlyAQ | 48 | Fly |
| 37 | FlyAR | 81 | Fly |

Query executed successfully.

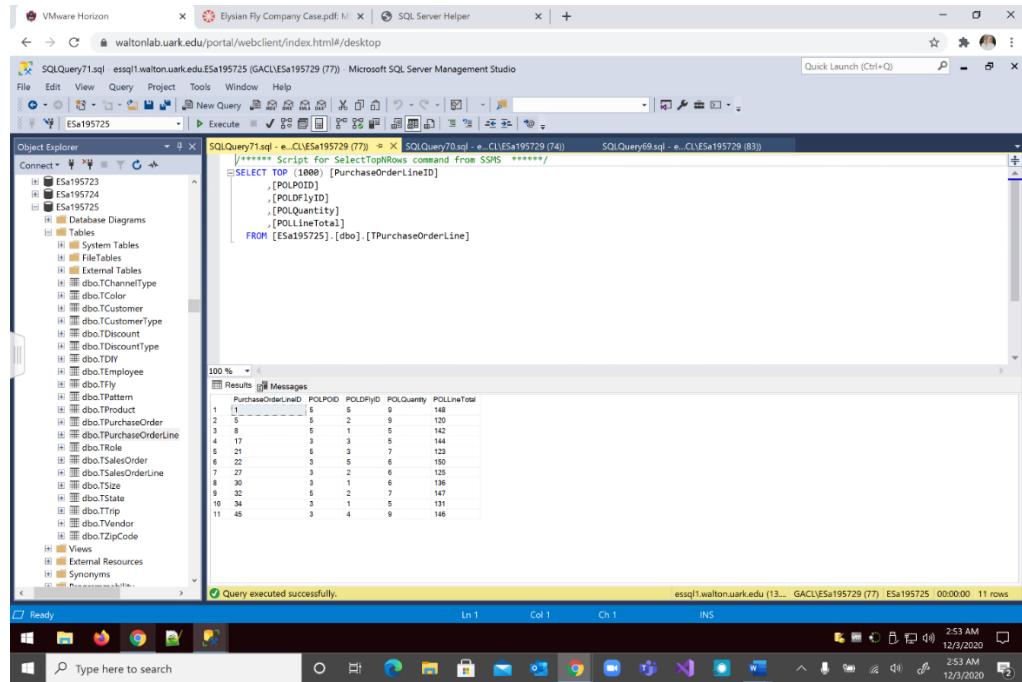
TPurchaseOrder:

The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including Es195723, Es195724, and Es195725. The current database is Es195725. The Results pane displays the output of a query on the TPurchaseOrder table, which contains 11 rows of purchase order information. The columns shown are PurchaseOrderID, POEmpID, PODate, and POTotal. The data includes various purchase orders with different dates and total amounts.

| PurchaseOrderID | POEmpID | PODate | POTotal |
|-----------------|---------|------------|---------|
| 1 | 3 | 2020-10-01 | 2600 |
| 2 | 5 | 2020-10-29 | 1000 |
| 3 | 7 | 2020-09-13 | 2700 |
| 4 | 9 | 2019-08-27 | 4000 |
| 5 | 10 | 2020-06-30 | 6500 |
| 6 | 11 | 2019-02-20 | 4700 |
| 7 | 12 | 2019-01-24 | 2400 |
| 8 | 13 | 2020-11-11 | 5600 |
| 9 | 14 | 2019-12-11 | 2200 |
| 10 | 15 | 2020-04-04 | 3200 |
| 11 | 17 | 2019-03-18 | 1700 |

Query executed successfully.

TPurchaseOrderLine:

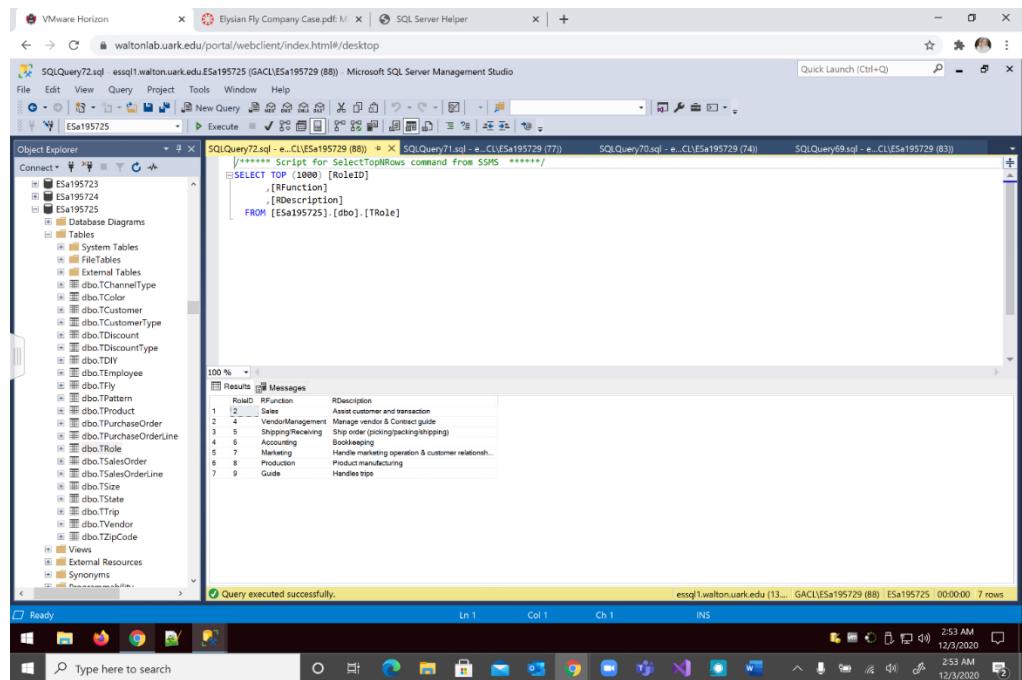


The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including Esal195723, Esal195724, Esal195725, and Esal195726. The central pane displays a T-SQL script for selecting the top 1000 rows from the TPurchaseOrderLine table. The results pane shows 11 rows of data with columns PurchaseOrderLineID, POLOD, POLODRefID, POQuantity, and POLLineTotal. The bottom status bar indicates the query was executed successfully at 12:3/2020 2:53 AM.

```

SELECT TOP (1000) [PurchaseOrderLineID]
      ,[POLOD]
      ,[POLODRefID]
      ,[POQuantity]
      ,[POLLineTotal]
  FROM [Esal195725].[dbo].[TPurchaseOrderLine]
  
```

TRole:

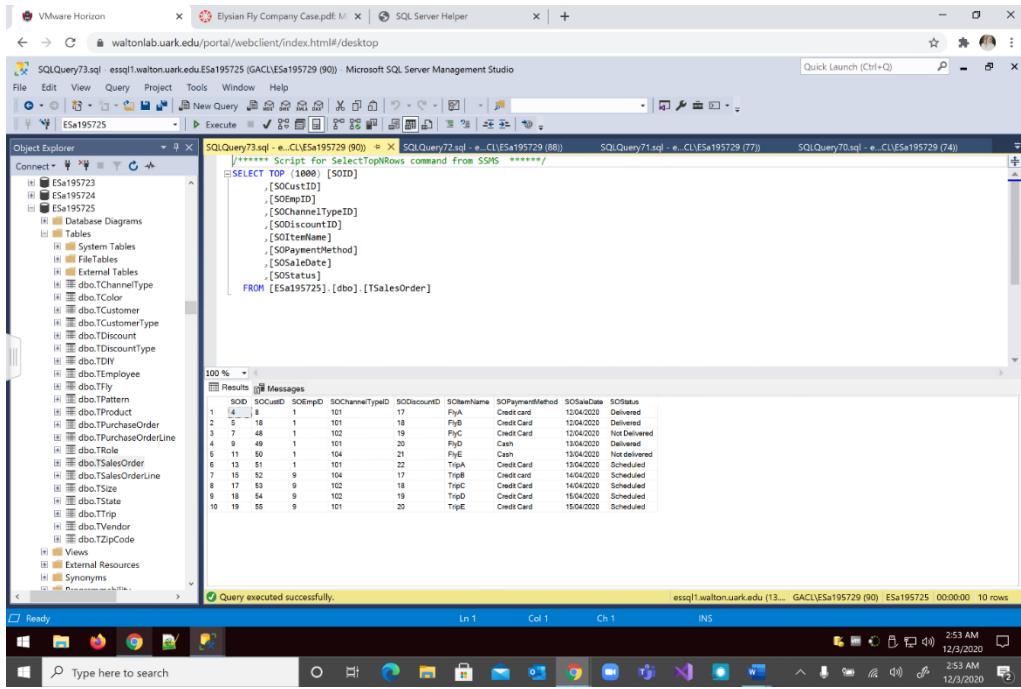


The screenshot shows the Microsoft SQL Server Management Studio interface. The Object Explorer on the left lists several databases, including Esal195723, Esal195724, Esal195725, and Esal195726. The central pane displays a T-SQL script for selecting the top 1000 rows from the TRole table. The results pane shows 7 rows of data with columns RoleID, RFunction, and RDescription. The bottom status bar indicates the query was executed successfully at 12/3/2020 2:53 AM.

```

SELECT TOP (1000) [RoleID]
      ,[RFunction]
      ,[RDescription]
  FROM [Esal195725].[dbo].[TRole]
  
```

TSalesOrder:

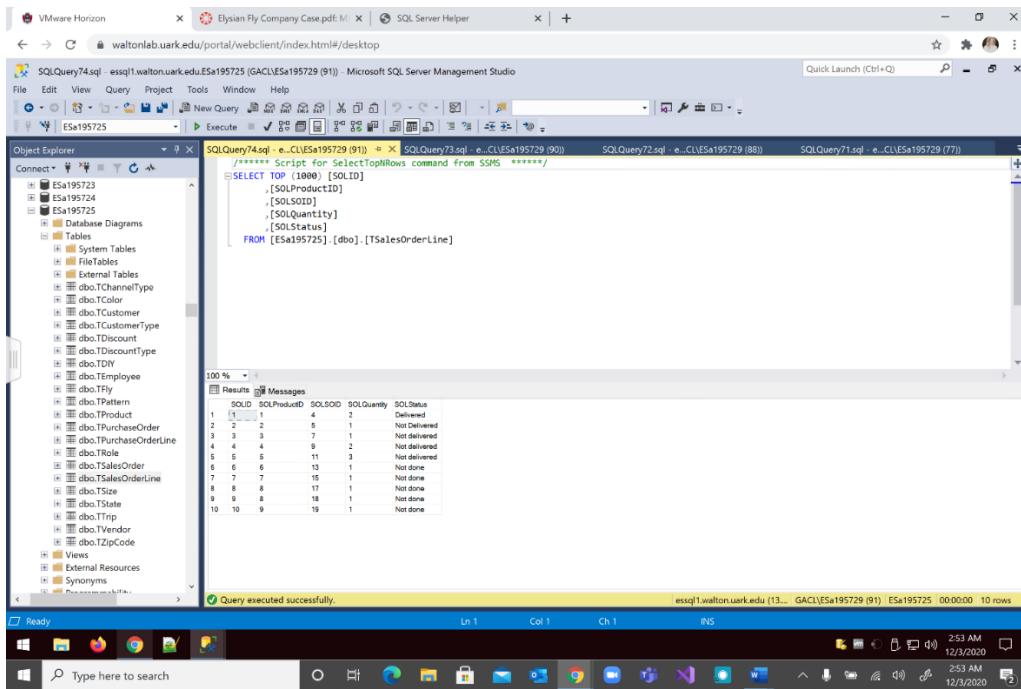


The screenshot shows the Microsoft SQL Server Management Studio interface. A query window is open with the following SQL code:

```
SELECT TOP (1000) [SOID]
      ,[SOcustID]
      ,[SOempID]
      ,[SOChannelTypeID]
      ,[SOdiscountID]
      ,[SOitemname]
      ,[SOPaymentMethod]
      ,[SOSaleDate]
      ,[SOSatus]
   FROM [Esa195725].[dbo].[TSalesOrder]
```

The results pane displays 10 rows of data from the TSalesOrder table. The columns are: SOID, SOcustID, SOempID, SOChannelTypeID, SOdiscountID, SOitemname, SOPaymentMethod, SOSaleDate, and SOSatus. The data includes various sales records with details like payment method (Credit Card, FlyB, FlyE), sale date (12/04/2020 to 15/04/2020), and status (Delivered, Not Delivered, Scheduled).

TSalesOrderLine:



The screenshot shows the Microsoft SQL Server Management Studio interface. A query window is open with the following SQL code:

```
SELECT TOP (1000) [SOLID]
      ,[SOLProductID]
      ,[SOLSOID]
      ,[SOLQuantity]
      ,[SOLStatus]
   FROM [Esa195725].[dbo].[TSalesOrderLine]
```

The results pane displays 10 rows of data from the TSalesOrderLine table. The columns are: SOLID, SLPProductID, SOLSOID, SOLQuantity, and SOLStatus. The data includes line items for various products with their respective quantities and statuses.

TSize:

| StandID | StateName |
|---------|-----------|
| 1 | AL |
| 2 | AK |
| 3 | AZ |
| 4 | AR |
| 5 | CA |
| 6 | CO |
| 7 | CT |
| 8 | DE |
| 9 | FL |
| 10 | GA |
| 11 | HI |
| 12 | ID |
| 13 | IL |
| 14 | IN |
| 15 | IA |
| 16 | KS |
| 17 | KY |
| 18 | LA |
| 19 | ME |
| 20 | MD |
| 21 | MA |
| 22 | MI |
| 23 | MN |
| 24 | MS |
| 25 | MO |
| 26 | MT |
| 27 | NE |
| 28 | NV |
| 29 | NH |
| 30 | NJ |
| 31 | NM |
| 32 | NV |
| 33 | NC |
| 34 | ND |
| 35 | OH |
| 36 | OK |
| 37 | OR |

TState:

| StandID | StateName |
|---------|-----------|
| 1 | AL |
| 2 | AK |
| 3 | AZ |
| 4 | AR |
| 5 | CA |
| 6 | CO |
| 7 | CT |
| 8 | DE |
| 9 | FL |
| 10 | GA |
| 11 | HI |
| 12 | ID |
| 13 | IL |
| 14 | IN |
| 15 | IA |
| 16 | KS |
| 17 | KY |
| 18 | LA |
| 19 | ME |
| 20 | MD |
| 21 | MA |
| 22 | MI |
| 23 | MN |
| 24 | MS |
| 25 | MO |
| 26 | MT |
| 27 | NE |
| 28 | NV |
| 29 | NH |
| 30 | NJ |
| 31 | NM |
| 32 | NV |
| 33 | NC |
| 34 | ND |
| 35 | OH |
| 36 | OK |
| 37 | OR |
| 38 | PA |
| 39 | RH |
| 40 | SD |
| 41 | TN |
| 42 | UT |
| 43 | VA |
| 44 | VI |
| 45 | WA |
| 46 | WV |
| 47 | WY |

TTrip:

The screenshot shows the Microsoft SQL Server Management Studio interface with four tabs open: SQLQuery77.sql, SQLQuery78.sql, SQLQuery75.sql, and SQLQuery74.sql. The TTrip table is displayed in the Results grid of the SQLQuery77.sql tab. The table has 20 rows and 13 columns: TripProductID, ProductID, Product, TripGUID, TripFishingSeason, TripDate, TripZipCodeID, TripWeather, TripBooked, TripNumOfDays, TripNumOfParticipants, TripRegulations, and TripGuidePreferences. The data includes various trip details like sunny or rainy weather, backpacking or kayaking preferences, and the number of participants.

| TripProductID | ProductID | Product | TripGUID | TripFishingSeason | TripDate | TripZipCodeID | TripWeather | TripBooked | TripNumOfDays | TripNumOfParticipants | TripRegulations | TripGuidePreferences |
|---------------|-----------|---------|----------|-------------------|------------|---------------|-------------|------------|---------------|-----------------------|-----------------|-----------------------------|
| 1 | 6 | 8 | 9 | Yes | 2020-04-13 | 22 | Sunny | Yes | 1 | 5 | None | |
| 2 | 3 | 7 | 10 | Yes | 2020-04-15 | 13 | Rainy | Yes | 1 | 5 | None | Backpacking |
| 3 | 2 | 8 | 48 | No | 2020-04-15 | 13 | Sunny | No | 1 | 10 | None | |
| 4 | 7 | 9 | 49 | No | 2020-04-17 | 41 | Sunny | No | 1 | 6 | None | |
| 5 | 8 | 10 | 50 | No | 2020-04-18 | 25 | Rainy | Yes | 1 | 2 | No Guests | Need to be good at kayaking |
| 6 | 9 | 7 | 51 | Yes | 2020-04-24 | 13 | Sunny | Yes | 1 | 5 | None | Need to be good at fishing |
| 7 | 10 | 6 | 52 | Yes | 2020-04-24 | 13 | Sunny | Yes | 1 | 7 | No Children | |
| 8 | 12 | 8 | 53 | Yes | 2020-05-02 | 10 | Windy | No | 2 | 10 | None | Cooking |
| 9 | 13 | 9 | 54 | Yes | 2020-05-04 | 13 | Sunny | Yes | 2 | 5 | None | |
| 10 | 14 | 10 | 55 | No | 2020-05-04 | 19 | Sunny | Yes | 1 | 10 | None | Kayaking |
| 11 | 23 | 5 | 56 | Yes | 2020-05-10 | 32 | Windy | Yes | 3 | 5 | None | |
| 12 | 25 | 9 | 18 | No | 2020-05-19 | 21 | Sunny | No | 3 | 2 | No Beginners | Cooking |
| 13 | 26 | 32 | 54 | Yes | 2020-12-18 | 3 | Rainy | No | 4 | 7 | None | |
| 14 | 28 | 36 | 24 | No | 2020-12-18 | 9 | Sunny | Yes | 3 | 8 | None | Backpacking |
| 15 | 30 | 19 | 290 | Yes | 2020-11-10 | 7 | Snowy | No | 3 | 10 | Experience | |
| 16 | 31 | 20 | 300 | Yes | 2020-05-01 | 3 | Windy | Yes | 1 | 9 | None | Kayaking |
| 17 | 36 | 9 | 53 | No | 2020-06-17 | 9 | Sunny | Yes | 8 | 1 | No Children | None |
| 18 | 37 | 12 | 21 | Yes | 2020-06-17 | 18 | Rainy | No | 2 | 1 | None | |
| 19 | 38 | 7 | 311 | Yes | 2020-09-08 | 8 | Sunny | Yes | 6 | 2 | None | Backpacking |
| 20 | 39 | 10 | 330 | No | 2020-11-14 | 18 | Snowy | No | 1 | 9 | No Children | Cooking |

TVendor:

The screenshot shows the Microsoft SQL Server Management Studio interface with four tabs open: SQLQuery78.sql, SQLQuery79.sql, SQLQuery76.sql, and SQLQuery75.sql. The TVendor table is displayed in the Results grid of the SQLQuery78.sql tab. The table has 10 rows and 13 columns: VendorID, VName, VVendorAccountID, VTerms, VZipCodeID, VTrialUsed, VDollarsSpentOnVendor, VPhone, VContactName, VFlyingTechniques, and VFlightTimeUsed. The data includes vendor names like Fels Punut Ac Ltd, Diam Corp, and Fly Brothers Co., along with their contact information and flying techniques.

| VendorID | VName | VVendorAccountID | VTerms | VZipCodeID | VTrialUsed | VDollarsSpentOnVendor | VPhone | VContactName | VFlyingTechniques |
|----------|-------------------|------------------|-------------|------------|------------|-----------------------|---------------|---------------|-------------------|
| 1 | Fels Punut Ac Ltd | 143 | 2/10 net 30 | 73 | 8 | 8676 | (360)555-4400 | James Tran | Double Knots |
| 2 | Gemini Corp | 276 | 3/10 net 30 | 12 | 8 | 10676 | (404)555-6100 | David Lee | Double Tie |
| 3 | Alpha Corp | 225 | 1/10 net 20 | 22 | 12 | 7056 | (346)551-5578 | Ella Roberts | Left Loop |
| 4 | Diam Corp | 138 | No discount | 34 | 3 | 4000 | (602)513-5862 | Hunter Eaving | Triple Bottom |
| 5 | Oscillation Id | 252 | 5/10 net 15 | 24 | 10 | 15000 | (107)542-6700 | Kyle Camey | Triple Tap |
| 6 | Delta Corp | 273 | No discount | 9 | 4 | 8000 | (108)542-5590 | Markus Bird | Round Head |
| 7 | Bikewind Corp | 312 | No discount | 2 | 6 | 6000 | (106)523-1231 | Iory Wang | Win's Hope |
| 8 | A Dollar Ltd | 127 | 2/10 net 30 | 13 | 13 | 9000 | (432)123-8241 | Noel Hunter | Around Neck |
| 9 | BeefFlies Co. | 736 | No discount | 86 | 26 | 17000 | (523)746-9837 | Jim Smith | Winner's Hope |
| 10 | Fly Brothers Co. | 896 | No discount | 98 | 52 | 18500 | (673)720-0887 | Tyler Prince | Pie-Synd Pech |

TZipCode:

| ZipCodeID | ZipCode | ZipCity |
|-----------|---------|----------------|
| 1 | 10001 | Adams |
| 2 | 10002 | Kleinrich |
| 3 | 10003 | Tierstra |
| 4 | 10004 | Helema |
| 5 | 10005 | Nicola |
| 6 | 10006 | Rodwell |
| 7 | 10007 | Merdorp |
| 8 | 10008 | Wageningen |
| 9 | 10009 | Zwanend |
| 10 | 10010 | Houtmanseel |
| 11 | 10011 | Crony-le-Graaf |
| 12 | 10012 | Bonvista |
| 13 | 10013 | Beeld |
| 14 | 10014 | SaintOnho |
| 15 | 10015 | Bloem |
| 16 | 10016 | Tarrasa |
| 17 | 10017 | Franeker |
| 18 | 10018 | Galea |
| 19 | 10019 | Musselburgh |
| 20 | 10020 | Haggeveen |
| 21 | 10021 | Oliver |
| 22 | 10022 | Furiosa |
| 23 | 10023 | Morenero |
| 24 | 10024 | Monterro Va. |
| 25 | 10025 | Limache |
| 26 | 10026 | Heidelberg |
| 27 | 10027 | Great Sudb |
| 28 | 10028 | Cher |
| 29 | 10029 | Bukelo |
| 30 | 10030 | Dernich |
| 31 | 10031 | Casteluccio S. |
| 32 | 10032 | Ramenskoye |
| 33 | 10033 | Ophrydeiem |
| 34 | 10034 | Amaria |
| 35 | 10035 | Lahore |
| 36 | 10036 | Sclayn |
| 37 | 10037 | Araka |
| 38 | 10038 | xanx |

Data Dictionary

A Data Dictionary is a set of names, data, ID numbers, emails, or any other kind of attribute that are being used in a database. The Data Dictionary helps to describe the data that is within the table and can also be used to list references, data types, primary and foreign keys, and more. Using it, users can organize their data before writing it in their code so that they can evaluate what type of data they will be using. The Data Dictionary can be important because it shows exactly how the data will be used. It also makes the data easier to understand and comprehend, by providing a concise table. Using the Data Dictionary has been helpful for the Elysian Fly Company Project because there is a long list of data that has to be executed, and the Data Dictionary table helps to organize the data that employees will need to reference. A complete Data Dictionary is listed below in the appendix.

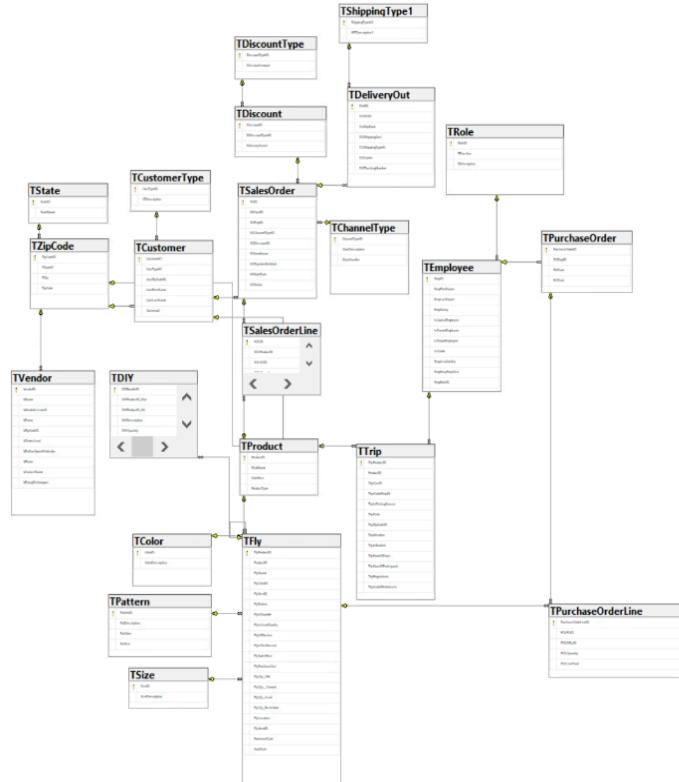
Denormalization

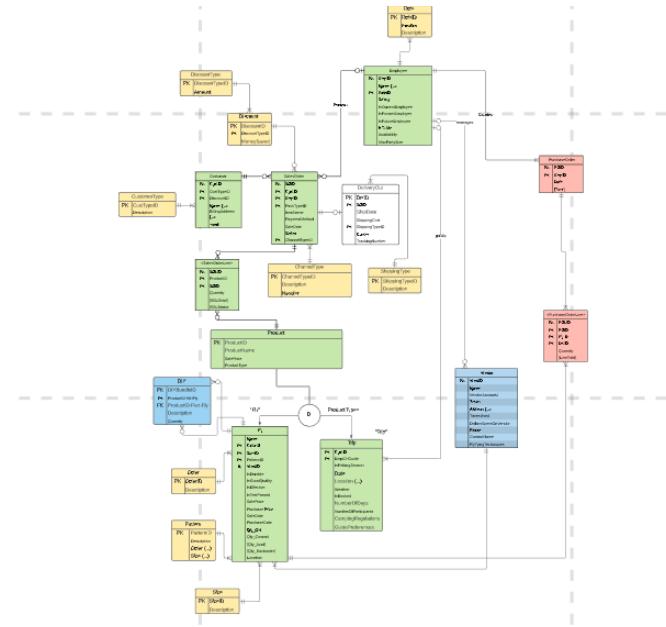
Denormalization is used to add data to one or more tables at one time. It helps users to ensure that their necessary data is entered and helps to efficiently add data to multiple tables in one statement, instead of having to enter the same data information multiple times. In this Elysian Fly Company Project, Team Mozart did not do much denormalization, because most of our tables did not pass the tests that are required before tables can go through the denormalization process. Another reason that Team Mozart did not use much denormalization is because currently, this is the primary database for Elysian Fly Company, and denormalization is typically used in the secondary database. For example, instead of adding Pattern to the TFly table, we preferred making it its own table. For a table to be denormalized, it must show two things: the table is referenced by only one table & one attribute of the table is used 90% of the

time. This is not the case for our TPattern table for example and that is why we did not denormalize it. However, we did add certain attributes to more than one table.

Implemented Physical Design

We came up with this diagram after implementing all our tables. It looks very similar to our ERD (see below).





Challenges Faced/Addressed During Implementation

One of the main challenges that Team Mozart had while creating the database was implementing the data into the tables. Using the data that Elysian Fly Company provided was helpful but having to create and implement the random data created some problems. To overcome this, we worked together as a team and helped one another to create and enter the data. We also did research and found ways to insert the data randomly and through other methods. Another challenge that Team Mozart faced was creating queries for Elysian Fly Company. While having to create additional queries that we thought Elysian Fly Company would be interested in seeing, we found it writing the code for the query to be challenging. To overcome this, we referred to videos and the textbook, and worked together. Working as a team while creating the queries helped to ensure that we were correctly writing them. Our last challenge that we faced would be creating the Data Dictionary. While very helpful when it is complete, the Data Dictionary can be difficult and time consuming to create. To overcome this, each team member created a Data Dictionary based on the tables they were assigned to create, and then we combined all of them. This helped because it lightened the workload on each member and helped each team member to gain experience creating a Data Dictionary. While we did have challenges, Team Mozart worked together to overcome them and do the best work that we could for Elysian Fly Company.

The challenges our team faced with creating new queries was the difficulty in knowing what the Elysian Fly Company specifically wanted out of the database and how to execute that need through our tables. In order to meet the needs of Elysian Fly Company we created queries that display the data from our ERD and logical design. Another challenge we faced was specifically implementing the queries to our database by creating three unique queries to our group for the Elysian Fly Company. We overcame this challenge by working as a group to think of queries that would benefit Elysian Fly Company in data that should be known and available to them. As a group we focused on areas that the customer was unaware of that they needed to be included into a database to make it as efficient and well-functioning to their needs and standards as a company. With that being said, our group included a multitude of tables and data that covers information dealing with their customer, product, and employees in order to create a well-rounded and efficient database. It

was hard to learn about all the complexities of SQL Server and areas dealing with the physical design of the database itself. We overcame this challenge by referencing lectures and notes, along with practicing SQL Server functions to become more acquainted.

Strengths and Weaknesses Encountered During Implementation

One of Team Mozart's strengths throughout this project has been our capability to get tasks completed, despite many obstacles being in the way. With this year looking so different, there were many challenges that got in the way of this project, but Team Mozart worked very hard to overcome those challenges and complete the project. Lam Nguyen, one of Team Mozart's members is currently in another time zone, and still proved to always be helpful and available no matter what. Having to meet primarily virtually and communicate through email, it was often hard to keep everything organized, but Team Mozart worked tirelessly together and overcame anything that was in the way. One of Team Mozart's weaknesses throughout this project was how different this past semester has looked. With the inability to do much in person, we had to find other ways to communicate and work on the project, but Team Mozart took full advantage of our resources and completed the project.

Specific SQL Statements Requested

Below are the queries that were requested by Elysian Fly.

| Query # | Question | SQL | Partial Output | | | | | | | | | | | | | | | | | | | | | | | | | |
|---------|---|---|---|--------|-----------|------------|------------|--------|----|------|---|---|----|------|------|---|----|------|---|------|---|---|---|---|------|---|---|---|
| 01 | Total Sales (in dollars) by customer state per year (e.g., total sales for all customers from Montana, Wyoming, Colorado, etc.) | <pre>SELECT StateName, Year(SOSaleDate) as SaleDate, COUNT (SOLID) AS TotalSales FROM TSalesOrderLine TSOL Join TSalesOrder TSO on TSOL.SOLID = TSO.SOID JOIN TCustomer AS TC ON TSO.SOCustID = TC.CustomerID JOIN TState AS TS ON TZC.ZStateID=TS.StateID WHERE (TS.StateName IN('MT', 'MN')) GROUP BY StateName, Year(SOSaleDate) ORDER BY Total Sales;</pre> | <table border="1"> <thead> <tr> <th></th> <th>StateName</th> <th>SaleDate</th> <th>TotalSales</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>MT</td> <td>2019</td> <td>1</td> </tr> <tr> <td>2</td> <td>MN</td> <td>2020</td> <td>1</td> </tr> <tr> <td>3</td> <td>MT</td> <td>2020</td> <td>1</td> </tr> </tbody> </table> | | StateName | SaleDate | TotalSales | 1 | MT | 2019 | 1 | 2 | MN | 2020 | 1 | 3 | MT | 2020 | 1 | | | | | | | | | |
| | StateName | SaleDate | TotalSales | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | MT | 2019 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | MN | 2020 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | MT | 2020 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | |
| 02 | Total sales (in dollars) by vendor per year. We must be able to calculate profit (sale price – purchase price). | <pre>SELECT year(SOSaleDate) as SaleDate, Count(SOLID) as TotalSales, VendorID, SUM(FlySalesPrice - FlyPurchaseCost) as Profit From TSalesOrder TS join TSalesOrderLine TSOL on TS.SOID = TSOL.SOLID join TProduct TP on TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID = TF.FlyProductID Join TVendor TV on TF.FlyVendID = TV.VendorID</pre> | <table border="1"> <thead> <tr> <th></th> <th>SaleDate</th> <th>TotalSales</th> <th>VendorID</th> <th>Profit</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>2020</td> <td>1</td> <td>1</td> <td>5</td> </tr> <tr> <td>2</td> <td>2019</td> <td>1</td> <td>3</td> <td>47</td> </tr> <tr> <td>3</td> <td>2020</td> <td>1</td> <td>6</td> <td>0</td> </tr> <tr> <td>4</td> <td>2020</td> <td>1</td> <td>7</td> <td>5</td> </tr> </tbody> </table> | | SaleDate | TotalSales | VendorID | Profit | 1 | 2020 | 1 | 1 | 5 | 2 | 2019 | 1 | 3 | 47 | 3 | 2020 | 1 | 6 | 0 | 4 | 2020 | 1 | 7 | 5 |
| | SaleDate | TotalSales | VendorID | Profit | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 2020 | 1 | 1 | 5 | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 2019 | 1 | 3 | 47 | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 2020 | 1 | 6 | 0 | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 2020 | 1 | 7 | 5 | | | | | | | | | | | | | | | | | | | | | | | | |

| | | GROUP BY year(SOSaleDate), VendorID; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|--|---|---|------------|--------------|------------|------------|------------|---------|----|----|-----|------|----|----|----|----|------|----|----|-----|---|------|---|----|-----|---|------|---|----|----|---|------|---|----|----|---|------|---|----|-----|-----|------|----|----|-----|----|------|----|----|-----|---|------|----|---|----|---|------|---|-----|----|---|---|----|---|---|----|---|---|
| 03 | The ten highest selling (in dollars) (a) patterns, (b) sizes, (c) pattern-size-color combinations in a given year. | <pre>SELECT top 10 FlyProductID, FlyPattern, Count(SOLID) as TotalSales, year(SOSaleDate) as SaleDate FROM TSalesOrder TS Join TSalesOrderLine TSOL on TS.SOID = TSOL.SOLID Join TProduct TP on TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID = TF.FlyProductID GROUP BY FlyProductID, FlyPattern, year(SOSaleDate) ORDER BY TotalSales desc, year(SOSaleDate) desc;</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>FlyProductID</th> <th>FlyPattern</th> <th>TotalSales</th> <th>SaleDate</th> </tr> </thead> <tbody> <tr><td>1</td><td>9</td><td>6</td><td>1</td><td>2021</td></tr> <tr><td>2</td><td>8</td><td>10</td><td>1</td><td>2021</td></tr> <tr><td>3</td><td>18</td><td>182</td><td>1</td><td>2021</td></tr> <tr><td>4</td><td>17</td><td>111</td><td>1</td><td>2021</td></tr> <tr><td>5</td><td>1</td><td>1</td><td>1</td><td>2021</td></tr> <tr><td>6</td><td>2</td><td>2</td><td>1</td><td>2020</td></tr> <tr><td>7</td><td>16</td><td>102</td><td>1</td><td>2020</td></tr> <tr><td>8</td><td>15</td><td>143</td><td>1</td><td>2020</td></tr> <tr><td>9</td><td>6</td><td>101</td><td>1</td><td>2019</td></tr> <tr><td>10</td><td>4</td><td>4</td><td>1</td><td>2019</td></tr> </tbody> </table> | | FlyProductID | FlyPattern | TotalSales | SaleDate | 1 | 9 | 6 | 1 | 2021 | 2 | 8 | 10 | 1 | 2021 | 3 | 18 | 182 | 1 | 2021 | 4 | 17 | 111 | 1 | 2021 | 5 | 1 | 1 | 1 | 2021 | 6 | 2 | 2 | 1 | 2020 | 7 | 16 | 102 | 1 | 2020 | 8 | 15 | 143 | 1 | 2020 | 9 | 6 | 101 | 1 | 2019 | 10 | 4 | 4 | 1 | 2019 | | | | | | | | | | | |
| | FlyProductID | FlyPattern | TotalSales | SaleDate | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 9 | 6 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 8 | 10 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 18 | 182 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 17 | 111 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 1 | 1 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 2 | 2 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 16 | 102 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 15 | 143 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 6 | 101 | 1 | 2019 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 4 | 4 | 1 | 2019 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | <pre>SELECT top 10 FlyProductID, FlySizeID, Count(SOLID) as TotalSales, year(SOSaleDate) as SaleDate FROM TSalesOrder TS Join TSalesOrderLine TSOL on TS.SOID = TSOL.SOLID Join TProduct TP on TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID = TF.FlyProductID GROUP BY FlyProductID, FlySizeID, year(SOSaleDate) ORDER BY TotalSales desc;</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>FlyProductID</th> <th>FlySizeID</th> <th>TotalSales</th> <th>SaleDate</th> </tr> </thead> <tbody> <tr><td>1</td><td>18</td><td>1</td><td>1</td><td>2021</td></tr> <tr><td>2</td><td>17</td><td>24</td><td>1</td><td>2021</td></tr> <tr><td>3</td><td>9</td><td>17</td><td>1</td><td>2021</td></tr> <tr><td>4</td><td>8</td><td>16</td><td>1</td><td>2021</td></tr> <tr><td>5</td><td>1</td><td>2</td><td>1</td><td>2021</td></tr> <tr><td>6</td><td>16</td><td>23</td><td>1</td><td>2020</td></tr> <tr><td>7</td><td>15</td><td>22</td><td>1</td><td>2020</td></tr> <tr><td>8</td><td>2</td><td>3</td><td>1</td><td>2020</td></tr> <tr><td>9</td><td>6</td><td>14</td><td>1</td><td>2019</td></tr> <tr><td>10</td><td>4</td><td>12</td><td>1</td><td>2019</td></tr> </tbody> </table> | | FlyProductID | FlySizeID | TotalSales | SaleDate | 1 | 18 | 1 | 1 | 2021 | 2 | 17 | 24 | 1 | 2021 | 3 | 9 | 17 | 1 | 2021 | 4 | 8 | 16 | 1 | 2021 | 5 | 1 | 2 | 1 | 2021 | 6 | 16 | 23 | 1 | 2020 | 7 | 15 | 22 | 1 | 2020 | 8 | 2 | 3 | 1 | 2020 | 9 | 6 | 14 | 1 | 2019 | 10 | 4 | 12 | 1 | 2019 | | | | | | | | | | | |
| | FlyProductID | FlySizeID | TotalSales | SaleDate | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 18 | 1 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 17 | 24 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 9 | 17 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 8 | 16 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 1 | 2 | 1 | 2021 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 16 | 23 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 15 | 22 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 2 | 3 | 1 | 2020 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 6 | 14 | 1 | 2019 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 4 | 12 | 1 | 2019 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | <pre>SELECT top 10 FlyProductID, FlyPattern, FlySizeID, FlyColorID, Count(SOLID) as TotalSales, year(SOSaleDate) as SaleDate FROM TSalesOrder TS Join TSalesOrderLine TSOL on TS.SOID = TSOL.SOLID Join TProduct TP on TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID = TF.FlyProductID GROUP BY FlyProductID, FlyPattern, FlySizeID, FlyColorID, year(SOSaleDate) ORDER BY TotalSales desc;</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>FlyProductID</th> <th>FlyPattern</th> <th>FlySizeID</th> <th>FlyColorID</th> <th>TotalSa</th> </tr> </thead> <tbody> <tr><td>1</td><td>18</td><td>182</td><td>1</td><td>17</td><td>1</td></tr> <tr><td>2</td><td>17</td><td>111</td><td>24</td><td>16</td><td>1</td></tr> <tr><td>3</td><td>9</td><td>6</td><td>17</td><td>9</td><td>1</td></tr> <tr><td>4</td><td>8</td><td>10</td><td>16</td><td>8</td><td>1</td></tr> <tr><td>5</td><td>1</td><td>1</td><td>2</td><td>1</td><td>1</td></tr> <tr><td>6</td><td>16</td><td>102</td><td>23</td><td>15</td><td>1</td></tr> <tr><td>7</td><td>15</td><td>143</td><td>22</td><td>14</td><td>1</td></tr> <tr><td>8</td><td>2</td><td>2</td><td>3</td><td>2</td><td>1</td></tr> <tr><td>9</td><td>6</td><td>101</td><td>14</td><td>6</td><td>1</td></tr> <tr><td>10</td><td>4</td><td>4</td><td>12</td><td>4</td><td>1</td></tr> </tbody> </table> | | FlyProductID | FlyPattern | FlySizeID | FlyColorID | TotalSa | 1 | 18 | 182 | 1 | 17 | 1 | 2 | 17 | 111 | 24 | 16 | 1 | 3 | 9 | 6 | 17 | 9 | 1 | 4 | 8 | 10 | 16 | 8 | 1 | 5 | 1 | 1 | 2 | 1 | 1 | 6 | 16 | 102 | 23 | 15 | 1 | 7 | 15 | 143 | 22 | 14 | 1 | 8 | 2 | 2 | 3 | 2 | 1 | 9 | 6 | 101 | 14 | 6 | 1 | 10 | 4 | 4 | 12 | 4 | 1 |
| | FlyProductID | FlyPattern | FlySizeID | FlyColorID | TotalSa | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 18 | 182 | 1 | 17 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 17 | 111 | 24 | 16 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 9 | 6 | 17 | 9 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 8 | 10 | 16 | 8 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 1 | 1 | 2 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 16 | 102 | 23 | 15 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 15 | 143 | 22 | 14 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 2 | 2 | 3 | 2 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 6 | 101 | 14 | 6 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 4 | 4 | 12 | 4 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| 04 | 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. | <pre>SELECT top 22 FlyProductID, Count(SOLID) as TotalSales FROM TSalesOrderLine TSOL full Join TProduct TP On TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID = TF.FlyProductID GROUP BY FlyProductID ORDER BY TotalSales desc;</pre> | <table border="1"> <thead> <tr> <th></th><th>FlyProductID</th><th>TotalSales</th></tr> </thead> <tbody> <tr><td>1</td><td>8</td><td>2</td></tr> <tr><td>2</td><td>9</td><td>1</td></tr> <tr><td>3</td><td>7</td><td>1</td></tr> <tr><td>4</td><td>6</td><td>1</td></tr> <tr><td>5</td><td>5</td><td>1</td></tr> <tr><td>6</td><td>4</td><td>1</td></tr> <tr><td>7</td><td>3</td><td>1</td></tr> <tr><td>8</td><td>2</td><td>1</td></tr> <tr><td>9</td><td>1</td><td>1</td></tr> <tr><td>10</td><td>22</td><td>0</td></tr> <tr><td>11</td><td>21</td><td>0</td></tr> <tr><td>12</td><td>20</td><td>0</td></tr> <tr><td>..</td><td>..</td><td>..</td></tr> </tbody> </table> | | FlyProductID | TotalSales | 1 | 8 | 2 | 2 | 9 | 1 | 3 | 7 | 1 | 4 | 6 | 1 | 5 | 5 | 1 | 6 | 4 | 1 | 7 | 3 | 1 | 8 | 2 | 1 | 9 | 1 | 1 | 10 | 22 | 0 | 11 | 21 | 0 | 12 | 20 | 0 | .. | .. | .. | | | | | | | | | | | | | |
|----|---|--|---|--------------------|--------------|--------------|-----------------|-------|---|----|----------|------|--------------------|---|----------|---|------|-------------|-----------|----|---|------|-----------|---|----|---|-----------|---------------|---|----|--------|------|---------------|----|--------|---|------|---------------|--------|----|----|------|--------------|----|----|---|------|-------------|---|---|---|------|--------------------|----|---|---|------|--------------------|
| | FlyProductID | TotalSales | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 8 | 2 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 9 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 7 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 6 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 5 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 4 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 3 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 2 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 1 | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 22 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | 21 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | 20 | 0 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| .. | .. | .. | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 05 | Total sales (in dollars) for each channel per month. | <pre>SELECT month(SOSaleDate) as SaleMonth, Count(SOLID) as TotalSales, ChanDescription FROM TSalesOrder TS join TSalesOrderLine TSOL ON TS.SOID = TSOL.SOLID Join TChannelType TCT on TS.SOChannelTypeID = TCT.ChannelTypeID GROUP BY month(SOSaleDate), ChanDescription;</pre> | <table border="1"> <thead> <tr> <th></th><th>SaleMonth</th><th>TotalSales</th><th>ChanDescription</th></tr> </thead> <tbody> <tr><td>1</td><td>4</td><td>1</td><td>By Phone</td></tr> <tr><td>2</td><td>10</td><td>1</td><td>By Phone</td></tr> <tr><td>3</td><td>1</td><td>1</td><td>In-Person</td></tr> <tr><td>4</td><td>3</td><td>2</td><td>In-Person</td></tr> <tr><td>5</td><td>11</td><td>1</td><td>In-Person</td></tr> <tr><td>6</td><td>2</td><td>1</td><td>Online</td></tr> <tr><td>7</td><td>3</td><td>1</td><td>Online</td></tr> <tr><td>8</td><td>4</td><td>1</td><td>Online</td></tr> <tr><td>9</td><td>7</td><td>1</td><td>Online</td></tr> </tbody> </table> | | SaleMonth | TotalSales | ChanDescription | 1 | 4 | 1 | By Phone | 2 | 10 | 1 | By Phone | 3 | 1 | 1 | In-Person | 4 | 3 | 2 | In-Person | 5 | 11 | 1 | In-Person | 6 | 2 | 1 | Online | 7 | 3 | 1 | Online | 8 | 4 | 1 | Online | 9 | 7 | 1 | Online | | | | | | | | | | | | | | | |
| | SaleMonth | TotalSales | ChanDescription | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 4 | 1 | By Phone | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 10 | 1 | By Phone | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 1 | 1 | In-Person | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 3 | 2 | In-Person | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 11 | 1 | In-Person | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 2 | 1 | Online | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 3 | 1 | Online | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 4 | 1 | Online | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 7 | 1 | Online | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 06 | 10% of the products that have the highest margin. | <pre>SELECT top 10(FlySalesPrice - FlyPurchaseCost) as profit, FlyproductID, FlyName, VName FROM TProduct TP join TFly TF on TP.ProductID = TF.FlyProductID join TVendor TV on TF.FlyVendorID = TV.VendorID ORDER BY profit desc;</pre> | <table border="1"> <thead> <tr> <th></th><th>profit</th><th>FlyproductID</th><th>FlyName</th><th>VName</th></tr> </thead> <tbody> <tr><td>1</td><td>78</td><td>10</td><td>FlyJ</td><td>Felis Purus Ac Ltd</td></tr> <tr><td>2</td><td>50</td><td>9</td><td>FlyI</td><td>Dave & Sons</td></tr> <tr><td>3</td><td>47</td><td>8</td><td>FlyH</td><td>Elit & Co</td></tr> <tr><td>4</td><td>43</td><td>6</td><td>FlyF</td><td>Sem Nulla Ltd</td></tr> <tr><td>5</td><td>39</td><td>21</td><td>FlyT</td><td>Sem Nulla Ltd</td></tr> <tr><td>6</td><td>10</td><td>2</td><td>FlyB</td><td>Sem Nulla Ltd</td></tr> <tr><td>7</td><td>10</td><td>3</td><td>FlyC</td><td>Octavian ltd</td></tr> <tr><td>8</td><td>5</td><td>4</td><td>FlyD</td><td>Dave & Sons</td></tr> <tr><td>9</td><td>5</td><td>5</td><td>FlyE</td><td>Felis Purus Ac Ltd</td></tr> <tr><td>10</td><td>5</td><td>1</td><td>FlyA</td><td>Felis Purus Ac Ltd</td></tr> </tbody> </table> | | profit | FlyproductID | FlyName | VName | 1 | 78 | 10 | FlyJ | Felis Purus Ac Ltd | 2 | 50 | 9 | FlyI | Dave & Sons | 3 | 47 | 8 | FlyH | Elit & Co | 4 | 43 | 6 | FlyF | Sem Nulla Ltd | 5 | 39 | 21 | FlyT | Sem Nulla Ltd | 6 | 10 | 2 | FlyB | Sem Nulla Ltd | 7 | 10 | 3 | FlyC | Octavian ltd | 8 | 5 | 4 | FlyD | Dave & Sons | 9 | 5 | 5 | FlyE | Felis Purus Ac Ltd | 10 | 5 | 1 | FlyA | Felis Purus Ac Ltd |
| | profit | FlyproductID | FlyName | VName | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 78 | 10 | FlyJ | Felis Purus Ac Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 50 | 9 | FlyI | Dave & Sons | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 47 | 8 | FlyH | Elit & Co | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 43 | 6 | FlyF | Sem Nulla Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 39 | 21 | FlyT | Sem Nulla Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 10 | 2 | FlyB | Sem Nulla Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 10 | 3 | FlyC | Octavian ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 5 | 4 | FlyD | Dave & Sons | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 5 | 5 | FlyE | Felis Purus Ac Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 5 | 1 | FlyA | Felis Purus Ac Ltd | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 07 | The ten most popular (units sold) DIY fly-tying materials. | <pre>SELECT FlyName, count(SOLID) as TotalSales FROM TSalesOrder TSO join TSalesOrderLine TSOL on TSO.SOID = TSOL.SOLID Join TProduct TP on TSOL.SOLProductID = TP.ProductID Join TFly TF on TP.ProductID =</pre> | <table border="1"> <thead> <tr> <th></th><th>FlyName</th><th>TotalSales</th></tr> </thead> <tbody> <tr><td></td><td></td><td></td></tr> </tbody> </table> <p>*No data found for this query</p> | | FlyName | TotalSales | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | FlyName | TotalSales | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

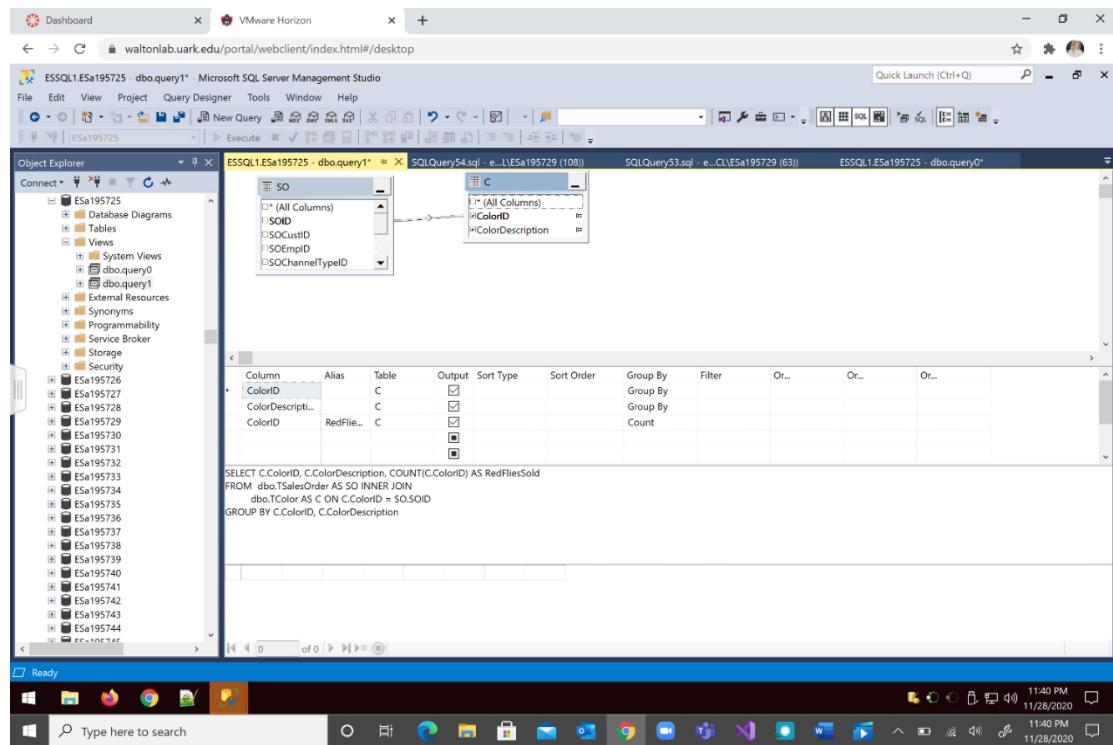
| | | TF.FlyProductID WHERE ProductType = 'bundle' GROUP BY FlyName; | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|----|--|---|--|-------------|-----------------------|--------------|--------------|-------------|-----------------------|---|----------|--------|---------|---------|------------|------|----------|--------|------|-------|------------|------|-------|------|-------|----------|------------|---|-------|------|-------|-------|------------|---|---|------|-------|----|------------|---|---|-------|--------|----|------------|---|----|------|-------|----|------------|
| 08 | The number of distinct products managed by each vendor manager. | <pre>Select Distinct EmpId, EmpFirstName, EmpLastName, COUNT(TProd.ProductID) AS ProductID From TVendor TVend JOIN TZipCode TZip ON TVend.VZipCodeID= TZip.ZipCodeID JOIN TCustomer TCust ON TZip.ZipCodeID = TCust.CustZipCodeID JOIN TProduct TProd ON TCust.CustomerID = TProd.ProductID JOIN TFly TF ON TProd.ProductID = TF.FlyProductID JOIN TTrip TTrip ON TProd.ProductID = TTrip.ProductID JOIN TEmployee Temp ON TTrip.TripGuideEmpID = Temp.EmpID Group By EmpID, EmpFirstName, EmpLastName Order By EmpFirstName</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>EmpID</th> <th>EmpFirstName</th> <th>EmpLastName</th> <th>ProductID</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>8</td> <td>Ryan</td> <td>Smith</td> <td>1</td> </tr> <tr> <td>2</td> <td>12</td> <td>Sean</td> <td>Thompson</td> <td>1</td> </tr> <tr> <td>3</td> <td>18</td> <td>Susan</td> <td>Nice</td> <td>1</td> </tr> </tbody> </table> | | EmpID | EmpFirstName | EmpLastName | ProductID | 1 | 8 | Ryan | Smith | 1 | 2 | 12 | Sean | Thompson | 1 | 3 | 18 | Susan | Nice | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | EmpID | EmpFirstName | EmpLastName | ProductID | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 8 | Ryan | Smith | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 12 | Sean | Thompson | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 18 | Susan | Nice | 1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 09 | 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, the number in the customer's party. | <pre>Select CustFirstName, CustLastName, EmpFirstName, EmpLastName, TripNumOfParticipants From TEmployee Temp JOIN TTrip TTrip ON Temp.EmpID = TTrip.TripProductID JOIN TProduct TProd ON TTrip.ProductID = TProd.ProductID JOIN TSalesOrderLine TSOL ON TProd.ProductID = TSOL.SOLProductID JOIN TSalesOrder TSO ON TSOL.SOLID = TSO.SOIID JOIN TCustomer TCust ON TSO.SOCustID = TCust.CustomerID Order By CustFirstName</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>CustFirstName</th> <th>CustLastName</th> <th>EmpFirstName</th> <th>EmpLastName</th> <th>TripNumOfParticipants</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>Courtney</td> <td>Hunter</td> <td>Tomma</td> <td>Bambara</td> <td>5</td> </tr> <tr> <td>2</td> <td>Courtney</td> <td>Hunter</td> <td>Jack</td> <td>Ruark</td> <td>7</td> </tr> <tr> <td>3</td> <td>James</td> <td>John</td> <td>Sean</td> <td>Thompson</td> <td>10</td> </tr> <tr> <td>4</td> <td>James</td> <td>John</td> <td>Laura</td> <td>Maure</td> <td>10</td> </tr> </tbody> </table> | | CustFirstName | CustLastName | EmpFirstName | EmpLastName | TripNumOfParticipants | 1 | Courtney | Hunter | Tomma | Bambara | 5 | 2 | Courtney | Hunter | Jack | Ruark | 7 | 3 | James | John | Sean | Thompson | 10 | 4 | James | John | Laura | Maure | 10 | | | | | | | | | | | | | | | | | | |
| | CustFirstName | CustLastName | EmpFirstName | EmpLastName | TripNumOfParticipants | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | Courtney | Hunter | Tomma | Bambara | 5 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Courtney | Hunter | Jack | Ruark | 7 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | James | John | Sean | Thompson | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | James | John | Laura | Maure | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | Number of trips and the number of customers taken on fishing trips by each guide in the past 6 months. | <pre>Select Distinct TripNumOfParticipants, EmpFirstName, EmpLastName, TripCustID, TripDate From TEmployee TEmp JOIN TTrip ON TEmp.EmpID = TTrip.TripProductID JOIN TCustomer TCust ON TTrip.TripCustID = TCust.CustomerID JOIN TSalesOrder TSO ON TCust.CustomerID = TSO.SOCustID Order By TripDate</pre> | <p>Results Messages</p> <table border="1"> <thead> <tr> <th></th> <th>TripNumOfParticipants</th> <th>EmpFirstName</th> <th>EmpLastName</th> <th>TripCustID</th> <th>TripDate</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>5</td> <td>Tomma</td> <td>Bambara</td> <td>18</td> <td>2020-04-14</td> </tr> <tr> <td>2</td> <td>6</td> <td>Sam</td> <td>Ly</td> <td>49</td> <td>2020-04-17</td> </tr> <tr> <td>3</td> <td>2</td> <td>Ryan</td> <td>Smith</td> <td>50</td> <td>2020-04-19</td> </tr> <tr> <td>4</td> <td>5</td> <td>Tony</td> <td>Lan</td> <td>51</td> <td>2020-04-21</td> </tr> <tr> <td>5</td> <td>7</td> <td>Jack</td> <td>Ruark</td> <td>52</td> <td>2020-04-24</td> </tr> <tr> <td>6</td> <td>5</td> <td>Katie</td> <td>Wright</td> <td>54</td> <td>2020-05-05</td> </tr> <tr> <td>7</td> <td>10</td> <td>Lucy</td> <td>Smith</td> <td>55</td> <td>2020-06-05</td> </tr> </tbody> </table> | | TripNumOfParticipants | EmpFirstName | EmpLastName | TripCustID | TripDate | 1 | 5 | Tomma | Bambara | 18 | 2020-04-14 | 2 | 6 | Sam | Ly | 49 | 2020-04-17 | 3 | 2 | Ryan | Smith | 50 | 2020-04-19 | 4 | 5 | Tony | Lan | 51 | 2020-04-21 | 5 | 7 | Jack | Ruark | 52 | 2020-04-24 | 6 | 5 | Katie | Wright | 54 | 2020-05-05 | 7 | 10 | Lucy | Smith | 55 | 2020-06-05 |
| | TripNumOfParticipants | EmpFirstName | EmpLastName | TripCustID | TripDate | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 5 | Tomma | Bambara | 18 | 2020-04-14 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 6 | Sam | Ly | 49 | 2020-04-17 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 2 | Ryan | Smith | 50 | 2020-04-19 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 5 | Tony | Lan | 51 | 2020-04-21 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 7 | Jack | Ruark | 52 | 2020-04-24 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 5 | Katie | Wright | 54 | 2020-05-05 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 10 | Lucy | Smith | 55 | 2020-06-05 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

| 11 | Names and email addresses of all customers who made purchases in a given month. | <pre>Select CustFirstName, CustLastName, CustEmail, SOSaleDate From TCustomer TCust Join TSalesOrder TSO ON TCust.CustomerID = TSO.SOCustID Order By SOSaleDate</pre> | <table border="1"> <thead> <tr> <th colspan="5">Results</th> </tr> <tr> <th></th> <th>CustFirstName</th> <th>CustLastName</th> <th>CustEmail</th> <th>SOSaleDate</th> </tr> </thead> <tbody> <tr><td>1</td><td>James</td><td>John</td><td>JamesJohn@gmail.com</td><td>2019-03-15 00:00:00.000</td></tr> <tr><td>2</td><td>Clark</td><td>Kent</td><td>Kent12@gmail.com</td><td>2019-07-30 00:00:00.000</td></tr> <tr><td>3</td><td>James</td><td>John</td><td>JamesJohn@gmail.com</td><td>2020-04-12 00:00:00.000</td></tr> <tr><td>4</td><td>Lam</td><td>Nguyen</td><td>Lam@ou.edu</td><td>2020-04-12 00:00:00.000</td></tr> <tr><td>5</td><td>Peter</td><td>Parker</td><td>PeterParker@ou.edu</td><td>2020-10-03 00:00:00.000</td></tr> <tr><td>6</td><td>Courtney</td><td>Hunter</td><td>Hunter.Courtney@gmail.com</td><td>2020-11-09 00:00:00.000</td></tr> <tr><td>7</td><td>Lam</td><td>Nguyen</td><td>Lam@ou.edu</td><td>2020-12-01 00:00:00.000</td></tr> <tr><td>8</td><td>James</td><td>John</td><td>JamesJohn@gmail.com</td><td>2020-12-20 00:00:00.000</td></tr> <tr><td>9</td><td>Courtney</td><td>Hunter</td><td>Hunter.Courtney@gmail.com</td><td>2021-01-13 00:00:00.000</td></tr> <tr><td>10</td><td>David</td><td>Black</td><td>Black.David78@gmail.com</td><td>2021-02-01 00:00:00.000</td></tr> <tr><td>11</td><td>Quinn</td><td>Sanders</td><td>odio.tristique@elicticum.edu</td><td>2021-03-15 00:00:00.000</td></tr> <tr><td>12</td><td>Bruce</td><td>Wayne</td><td>IAmBatmanlol@ou.edu</td><td>2021-04-12 00:00:00.000</td></tr> </tbody> </table> | Results | | | | | | CustFirstName | CustLastName | CustEmail | SOSaleDate | 1 | James | John | JamesJohn@gmail.com | 2019-03-15 00:00:00.000 | 2 | Clark | Kent | Kent12@gmail.com | 2019-07-30 00:00:00.000 | 3 | James | John | JamesJohn@gmail.com | 2020-04-12 00:00:00.000 | 4 | Lam | Nguyen | Lam@ou.edu | 2020-04-12 00:00:00.000 | 5 | Peter | Parker | PeterParker@ou.edu | 2020-10-03 00:00:00.000 | 6 | Courtney | Hunter | Hunter.Courtney@gmail.com | 2020-11-09 00:00:00.000 | 7 | Lam | Nguyen | Lam@ou.edu | 2020-12-01 00:00:00.000 | 8 | James | John | JamesJohn@gmail.com | 2020-12-20 00:00:00.000 | 9 | Courtney | Hunter | Hunter.Courtney@gmail.com | 2021-01-13 00:00:00.000 | 10 | David | Black | Black.David78@gmail.com | 2021-02-01 00:00:00.000 | 11 | Quinn | Sanders | odio.tristique@elicticum.edu | 2021-03-15 00:00:00.000 | 12 | Bruce | Wayne | IAmBatmanlol@ou.edu | 2021-04-12 00:00:00.000 |
|---------|---|--|---|-------------------------|--|--|--|--|------------|-----------------|--------------|-----------------------|--------------|---|-------|-----------|---------------------|-------------------------|-------|-------|-----------|------------------|-------------------------|---|-------|-----------|---------------------|-------------------------|----|-----|--------------|------------|-------------------------|---|-------|--------------|--------------------|-------------------------|---|----------|--------------|---------------------------|-------------------------|---|-----|--------|------------|-------------------------|---|-------|------|---------------------|-------------------------|---|----------|--------|---------------------------|-------------------------|----|-------|-------|-------------------------|-------------------------|----|-------|---------|------------------------------|-------------------------|----|-------|-------|---------------------|-------------------------|
| Results | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | CustFirstName | CustLastName | CustEmail | SOSaleDate | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | James | John | JamesJohn@gmail.com | 2019-03-15 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | Clark | Kent | Kent12@gmail.com | 2019-07-30 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | James | John | JamesJohn@gmail.com | 2020-04-12 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | Lam | Nguyen | Lam@ou.edu | 2020-04-12 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | Peter | Parker | PeterParker@ou.edu | 2020-10-03 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | Courtney | Hunter | Hunter.Courtney@gmail.com | 2020-11-09 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | Lam | Nguyen | Lam@ou.edu | 2020-12-01 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | James | John | JamesJohn@gmail.com | 2020-12-20 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | Courtney | Hunter | Hunter.Courtney@gmail.com | 2021-01-13 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | David | Black | Black.David78@gmail.com | 2021-02-01 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | Quinn | Sanders | odio.tristique@elicticum.edu | 2021-03-15 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | Bruce | Wayne | IAmBatmanlol@ou.edu | 2021-04-12 00:00:00.000 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | Number of times used, and dollars spent on each shipping vendor and shipping type by vendor. | <pre>Select ShippintTypeID1, VName, VDollarsSpentOnVendor, VTimesUsed From TShippingType1 TShipTyp JOIN TDeliveryOut TDO ON TShipTyp.ShippingTypeID1 = TDO.DOShippingTypeID Join TSalesOrder TSO ON TDO.DOSOID = TSO.SOVID JOIN TCustomer TCust ON TSO.SOCustID = TCust.CustomerID JOIN TZipCode TZ ON TCust.CustZipCodeID = TZ.ZipCodeID JOIN TVendor TVend ON TZ.ZipCodeID = TVend.VZipCodeID Order By VName</pre> | <table border="1"> <thead> <tr> <th colspan="5">Results</th> </tr> <tr> <th></th> <th>ShippingTypeID1</th> <th>VName</th> <th>VDollarsSpentOnVendor</th> <th>VTimesUsed</th> </tr> </thead> <tbody> <tr><td>1</td><td>1</td><td>Elit & Co</td><td>7500</td><td>12</td></tr> <tr><td>2</td><td>2</td><td>Elit & Co</td><td>7500</td><td>12</td></tr> <tr><td>3</td><td>3</td><td>Elit & Co</td><td>7500</td><td>12</td></tr> <tr><td>4</td><td>1</td><td>Octavian ltd</td><td>15000</td><td>10</td></tr> <tr><td>5</td><td>2</td><td>Octavian ltd</td><td>15000</td><td>10</td></tr> <tr><td>6</td><td>3</td><td>Octavian ltd</td><td>15000</td><td>10</td></tr> </tbody> </table> | Results | | | | | | ShippingTypeID1 | VName | VDollarsSpentOnVendor | VTimesUsed | 1 | 1 | Elit & Co | 7500 | 12 | 2 | 2 | Elit & Co | 7500 | 12 | 3 | 3 | Elit & Co | 7500 | 12 | 4 | 1 | Octavian ltd | 15000 | 10 | 5 | 2 | Octavian ltd | 15000 | 10 | 6 | 3 | Octavian ltd | 15000 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Results | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | ShippingTypeID1 | VName | VDollarsSpentOnVendor | VTimesUsed | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 1 | Elit & Co | 7500 | 12 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 2 | Elit & Co | 7500 | 12 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 3 | Elit & Co | 7500 | 12 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 1 | Octavian ltd | 15000 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 2 | Octavian ltd | 15000 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 3 | Octavian ltd | 15000 | 10 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 13 | Invoice lines for a given sales invoice number and given customer name. | <pre>Select SOCustID, SOLProductID, CustFirstName, CustLastName, SOLSOID From TCustomer TCust Join TSalesOrder TSO ON TCust.CustomerID = TSO.SOVID JOIN TSalesOrderLine TSOL ON TSO.SOVID = TSOL.SOLSOID JOIN TProduct TProd ON TSOL.SOLProductID = TProd.ProductID Order By SOCustID</pre> | <table border="1"> <thead> <tr> <th colspan="5">Results</th> </tr> <tr> <th></th> <th>SOCustID</th> <th>SOLProductID</th> <th>CustFirstName</th> <th>CustLastName</th> </tr> </thead> <tbody> <tr><td>1</td><td>18</td><td>2</td><td>Lam</td><td>Nguyen</td></tr> <tr><td>2</td><td>18</td><td>96</td><td>Lam</td><td>Nguyen</td></tr> <tr><td>3</td><td>18</td><td>99</td><td>Lam</td><td>Nguyen</td></tr> <tr><td>4</td><td>49</td><td>4</td><td>James</td><td>John</td></tr> <tr><td>5</td><td>50</td><td>5</td><td>Peter</td><td>Parker</td></tr> <tr><td>6</td><td>51</td><td>6</td><td>Clark</td><td>Kent</td></tr> <tr><td>7</td><td>52</td><td>7</td><td>Bruce</td><td>Wayne</td></tr> <tr><td>8</td><td>54</td><td>3</td><td>Courtney</td><td>Hunter</td></tr> <tr><td>9</td><td>54</td><td>8</td><td>Courtney</td><td>Hunter</td></tr> <tr><td>10</td><td>55</td><td>9</td><td>David</td><td>Black</td></tr> <tr><td>11</td><td>288</td><td>1</td><td>Quinn</td><td>Sanders</td></tr> <tr><td>12</td><td>288</td><td>8</td><td>Quinn</td><td>Sanders</td></tr> </tbody> </table> | Results | | | | | | SOCustID | SOLProductID | CustFirstName | CustLastName | 1 | 18 | 2 | Lam | Nguyen | 2 | 18 | 96 | Lam | Nguyen | 3 | 18 | 99 | Lam | Nguyen | 4 | 49 | 4 | James | John | 5 | 50 | 5 | Peter | Parker | 6 | 51 | 6 | Clark | Kent | 7 | 52 | 7 | Bruce | Wayne | 8 | 54 | 3 | Courtney | Hunter | 9 | 54 | 8 | Courtney | Hunter | 10 | 55 | 9 | David | Black | 11 | 288 | 1 | Quinn | Sanders | 12 | 288 | 8 | Quinn | Sanders |
| Results | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | SOCustID | SOLProductID | CustFirstName | CustLastName | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 18 | 2 | Lam | Nguyen | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 18 | 96 | Lam | Nguyen | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 18 | 99 | Lam | Nguyen | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 49 | 4 | James | John | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 50 | 5 | Peter | Parker | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 51 | 6 | Clark | Kent | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 7 | 52 | 7 | Bruce | Wayne | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 8 | 54 | 3 | Courtney | Hunter | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 9 | 54 | 8 | Courtney | Hunter | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 10 | 55 | 9 | David | Black | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 11 | 288 | 1 | Quinn | Sanders | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 12 | 288 | 8 | Quinn | Sanders | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 14 | 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 used the discount. | <pre>Select Distinct DiscountID, DDiscountTypeID, DMoneySaved From TDiscount TDis Join TSalesOrder TSO ON TDis.DiscountID = TSO.SODiscountID JOIN TSalesOrderLine TSOL ON TSO.SOVID = TSOL.SOLSOID JOIN TProduct TProd ON TSOL.SOLProductID = TProd.ProductID Order By DDiscountTypeID</pre> | <table border="1"> <thead> <tr> <th colspan="4">Results</th> </tr> <tr> <th></th> <th>DiscountID</th> <th>DDiscountTypeID</th> <th>DMoneySaved</th> </tr> </thead> <tbody> <tr><td>1</td><td>17</td><td>1</td><td>64.00</td></tr> <tr><td>2</td><td>18</td><td>2</td><td>97.00</td></tr> <tr><td>3</td><td>19</td><td>3</td><td>6.00</td></tr> <tr><td>4</td><td>20</td><td>4</td><td>17.00</td></tr> <tr><td>5</td><td>21</td><td>5</td><td>49.00</td></tr> <tr><td>6</td><td>22</td><td>6</td><td>28.00</td></tr> </tbody> </table> | Results | | | | | DiscountID | DDiscountTypeID | DMoneySaved | 1 | 17 | 1 | 64.00 | 2 | 18 | 2 | 97.00 | 3 | 19 | 3 | 6.00 | 4 | 20 | 4 | 17.00 | 5 | 21 | 5 | 49.00 | 6 | 22 | 6 | 28.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Results | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| | DiscountID | DDiscountTypeID | DMoneySaved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 1 | 17 | 1 | 64.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 2 | 18 | 2 | 97.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 3 | 19 | 3 | 6.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 4 | 20 | 4 | 17.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5 | 21 | 5 | 49.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 6 | 22 | 6 | 28.00 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

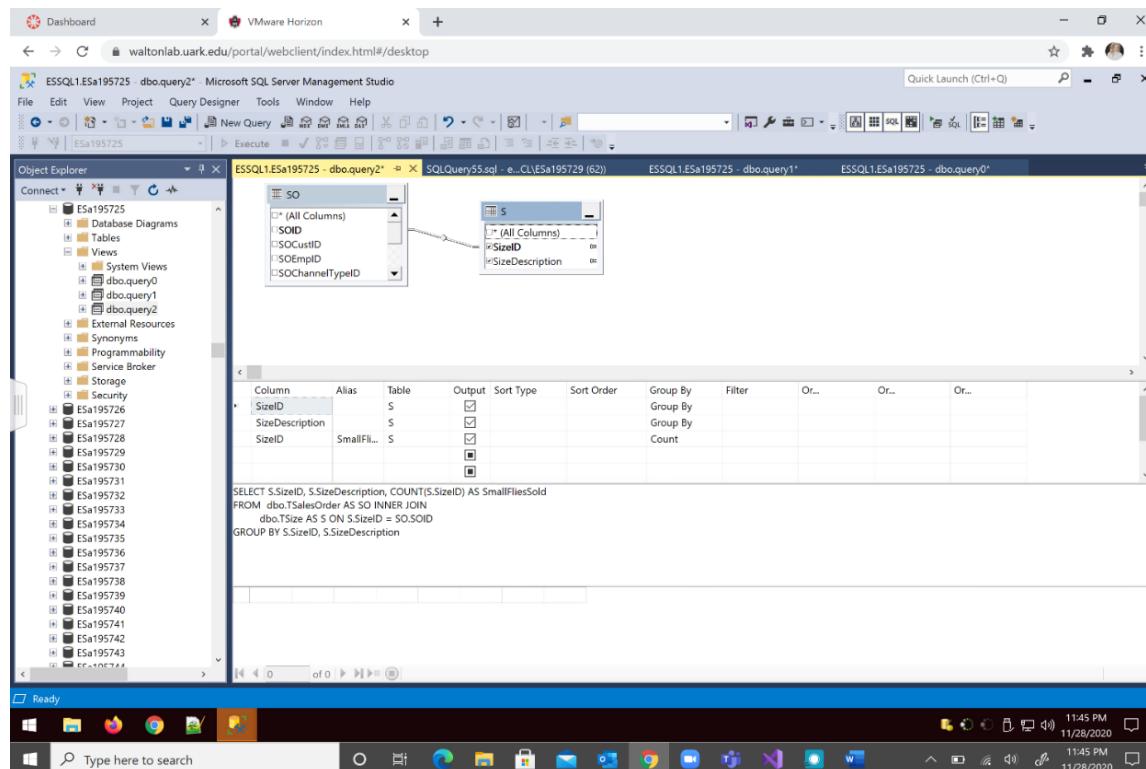
Three Additional Queries

| Query Number | Question | Importance | SQL | Partial Output | Recap of Findings |
|--------------|---------------------------------|--|---|--------------------------------|----------------------------------|
| 1 | How many red flies are sold? | This is important because Elysian Fly Company will be able to consider what color fly is most popular. | <pre>SELECT C.ColorID, C.ColorDescription, COUNT(C.ColorID) AS RedFliesSold FROM dbo.TSalesOrder AS SO INNER JOIN dbo.TColor as C ON C.ColorID = SO.SOID GROUP BY C.ColorID, C.ColorDescription</pre> | (See below on Query 1 Picture) | Only one Red Fly was sold. |
| 2 | How many small flies were sold? | Elysian Fly might need to know the number and percentage of each size fly that sells. | <pre>SELECT S.SizeID, S.SizeDescription, COUNT(S.SizeID) AS SmallFliesSold FROM dbo.TSalesOrder AS SO INNER JOIN dbo.TSize AS S ON S.SizeID = SO.SOID</pre> | (See below on Query 2 Picture) | 22 Small Flies were sold. |
| 3 | Customer who is Local | By finding this information, Elysian Fly Company will be able to tell if more locals buy products than customers that are not local. | <pre>SELECT * FROM TCustomer TC JOIN TCustomerType TCT on TC.CustTypeID = TCT.CustTypeID WHERE CTDescription = 'Local'</pre> | (See below on Query 3 Picture) | There were many local customers. |

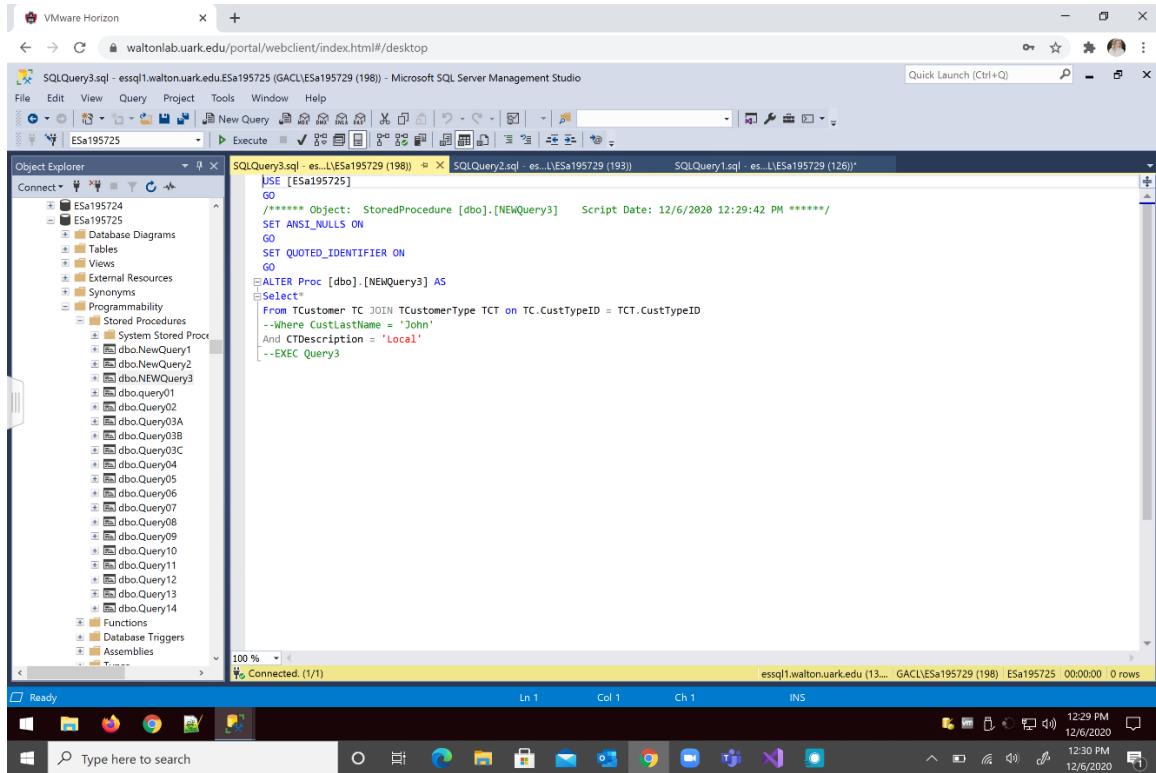
Query 1 Picture:



Query 2 Picture:



Query 3 Picture:



User Documentation

To use the database we created, please refer the following steps. We hope that it will ease the experience of our customer.

Step 1:

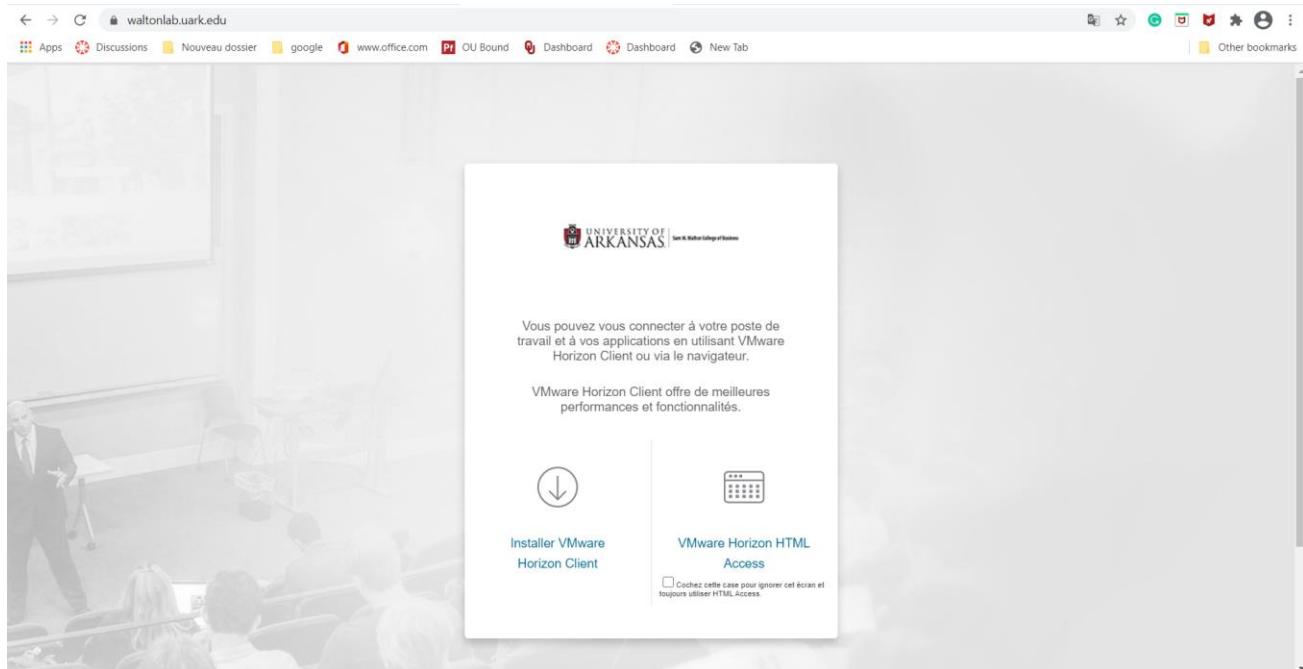
Log in to your computer.

Step 2:

Our database is implemented in a virtual environment offered to our team by the University of Arkansas. To access it, please click on the link below:

<https://waltonlab.uark.edu/>

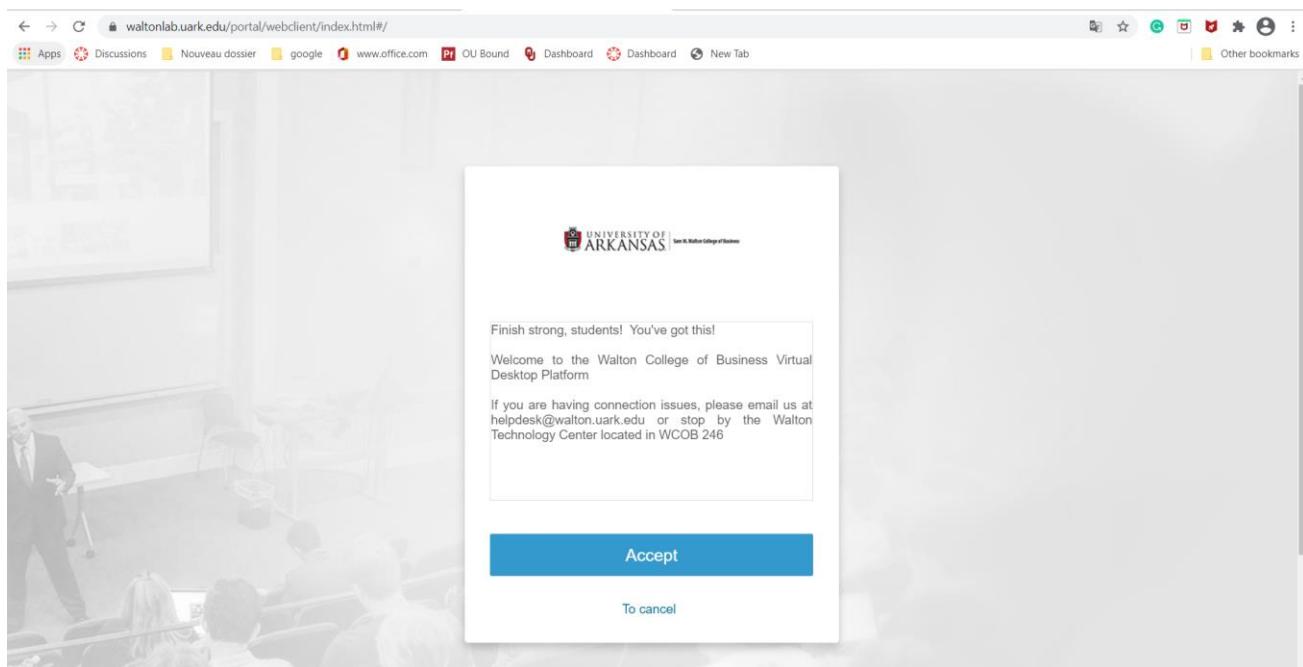
You should be directed to the following page.



Step 3:

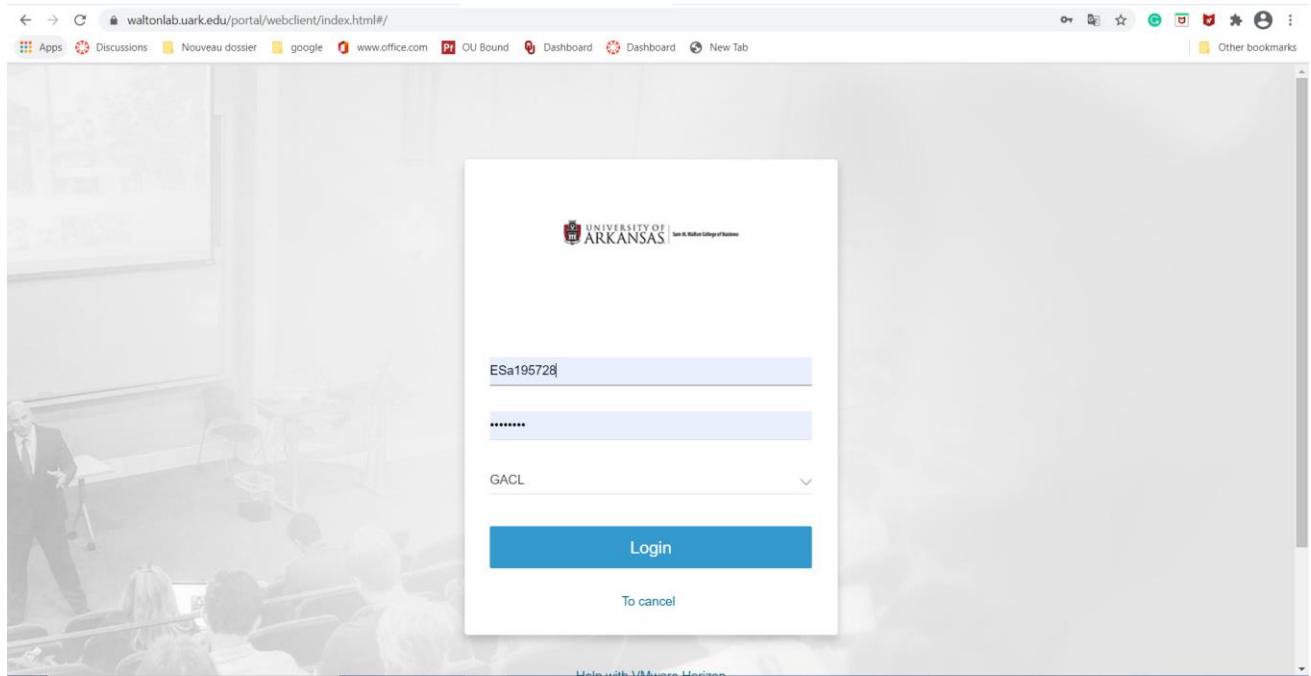
Click on **VMware Horizon HTML Access**.

You should be directed to the following page.



Click on **Accept**.

You should be directed to this page.



Step 4:

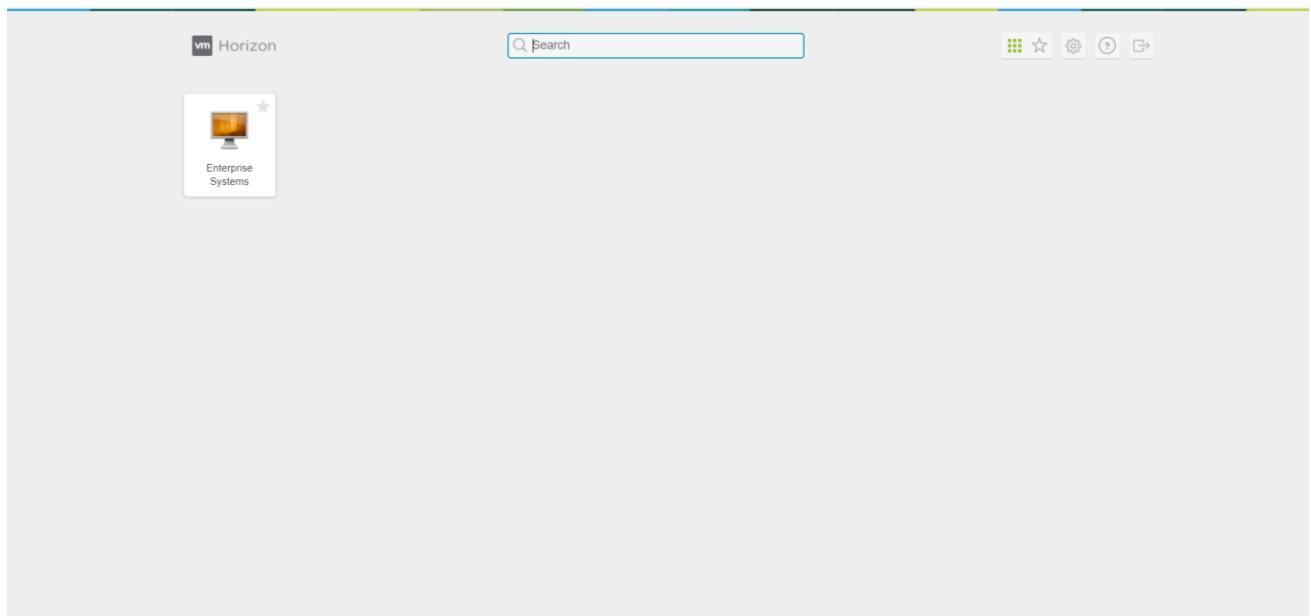
Enter the following:

Username: Esa195728

Password: RK96nsm\$

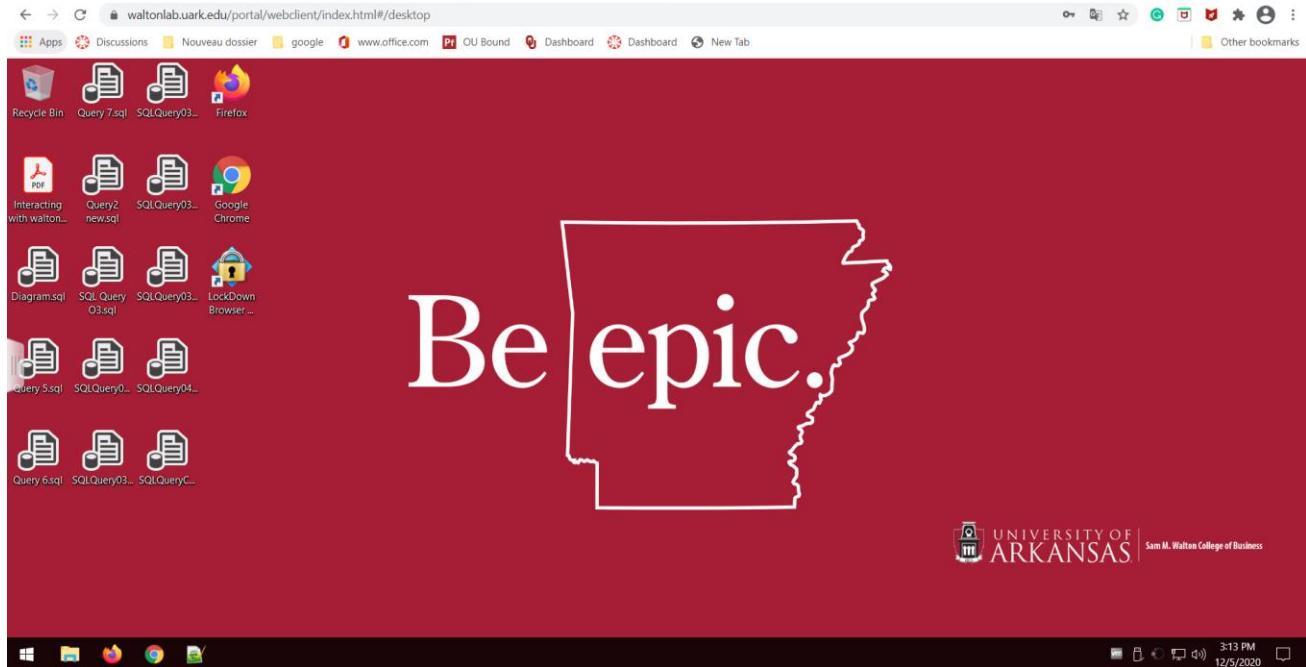
Then click on **Login**.

You should be directed to this page.



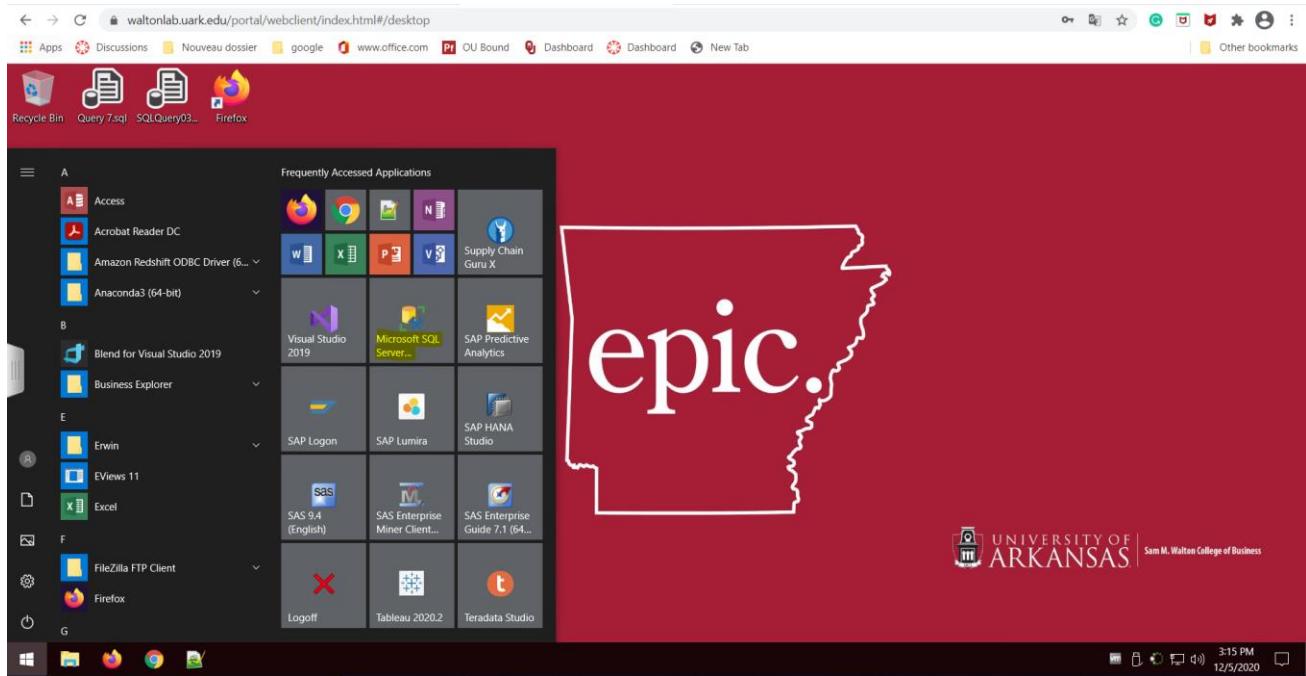
Step 5:

Click on **Enterprise Systems**. This will direct you to the windows virtual environment.



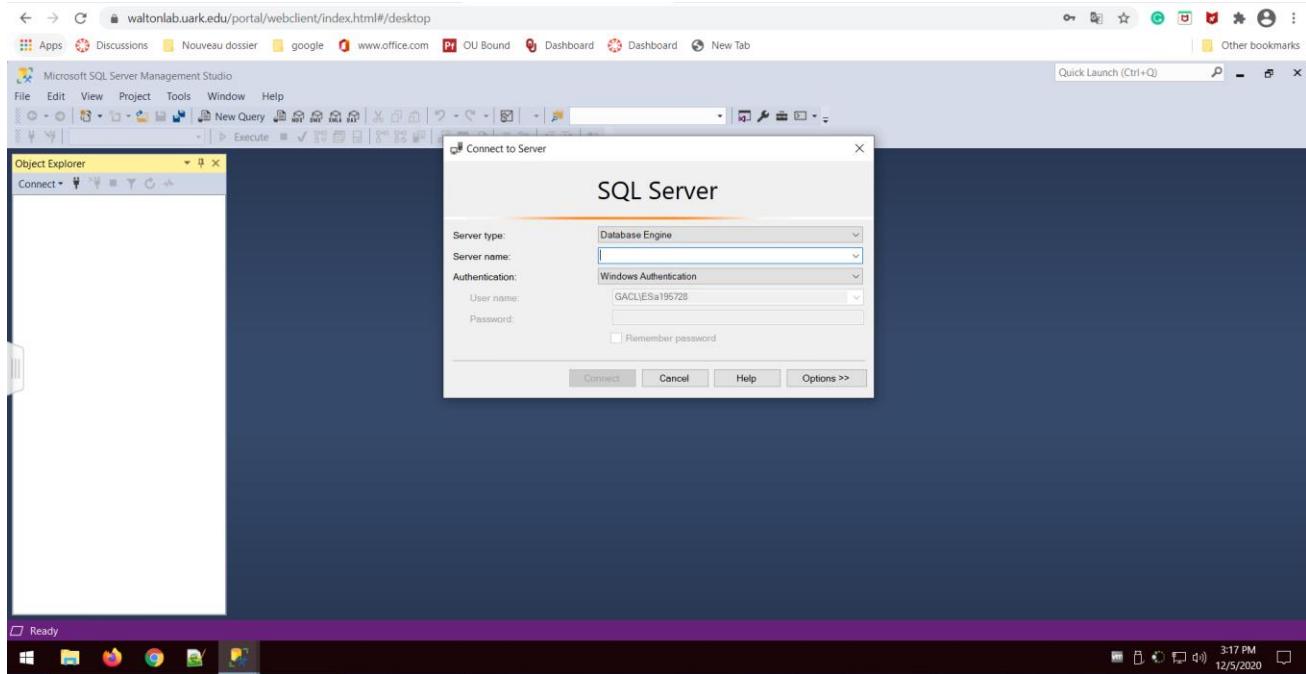
Step 6:

Click on **Start**.



Choose **Microsoft SQL Server** highlighted in yellow. This will open the Microsoft SQL Management studio.

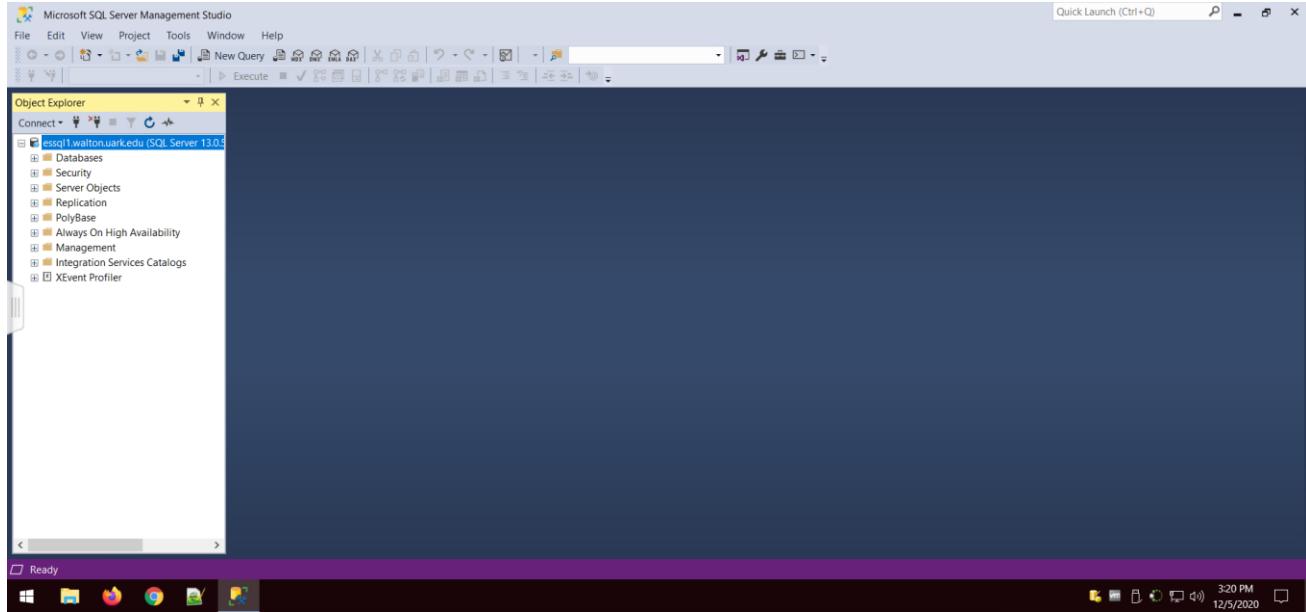
You should get the following tab.



To connect to the SQL server enter the following: **essql1.walton.uark.edu**

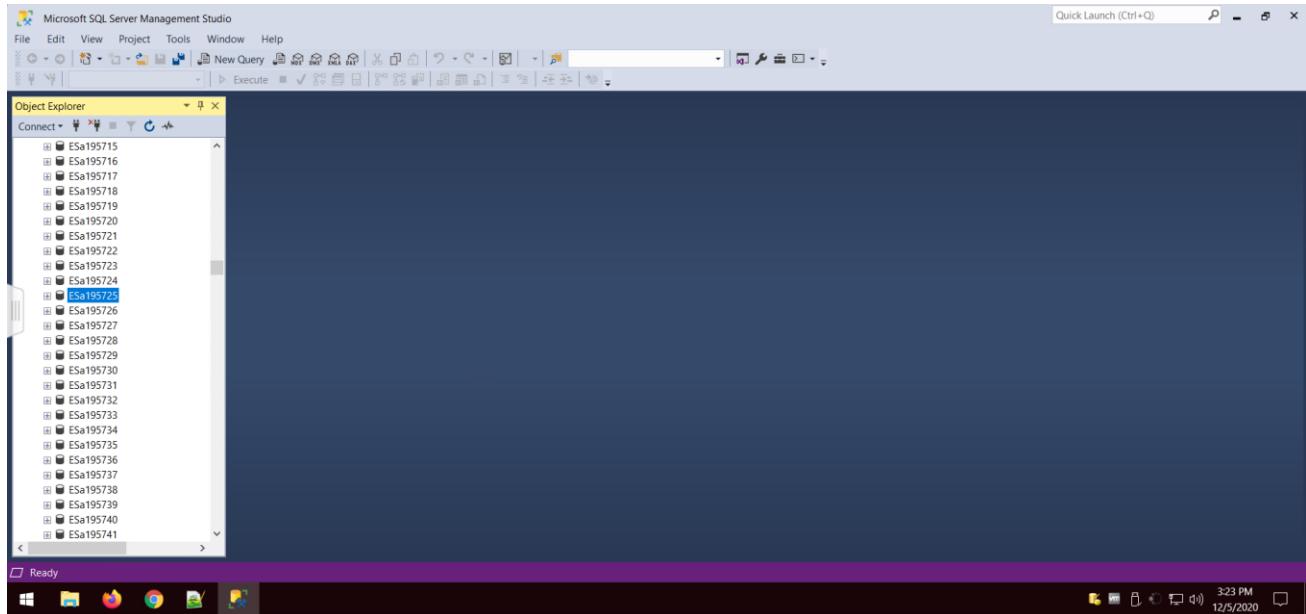
Then click on **connect**.

You should be directed to the page.

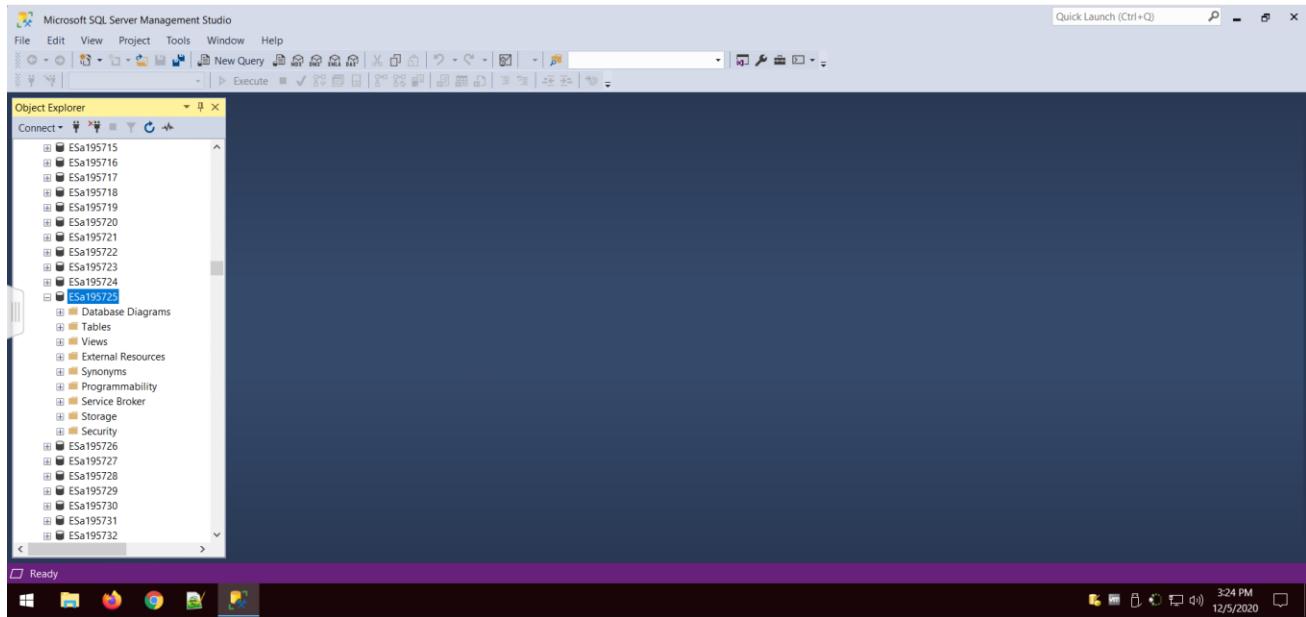


Step 7:

To open the database, click on the + sign next to Databases. This will open a list of databases that are stored in this server. Scroll down to find our Database (**Esa195725**).

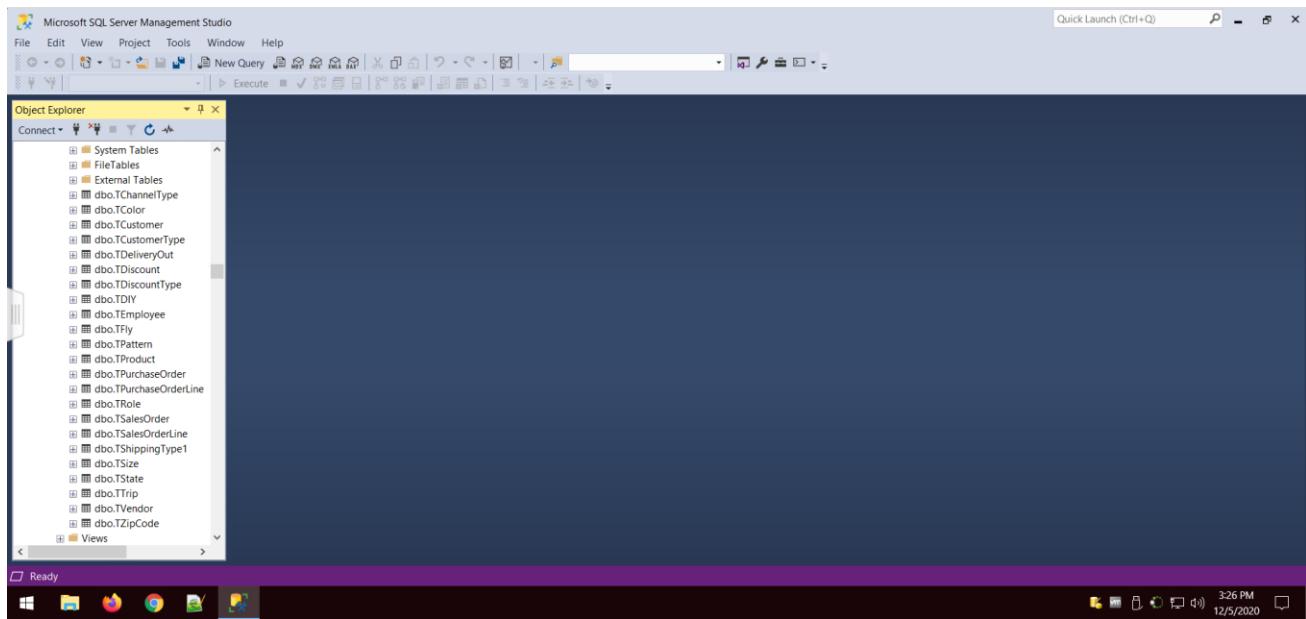


Click on the + sign next to it to expand.



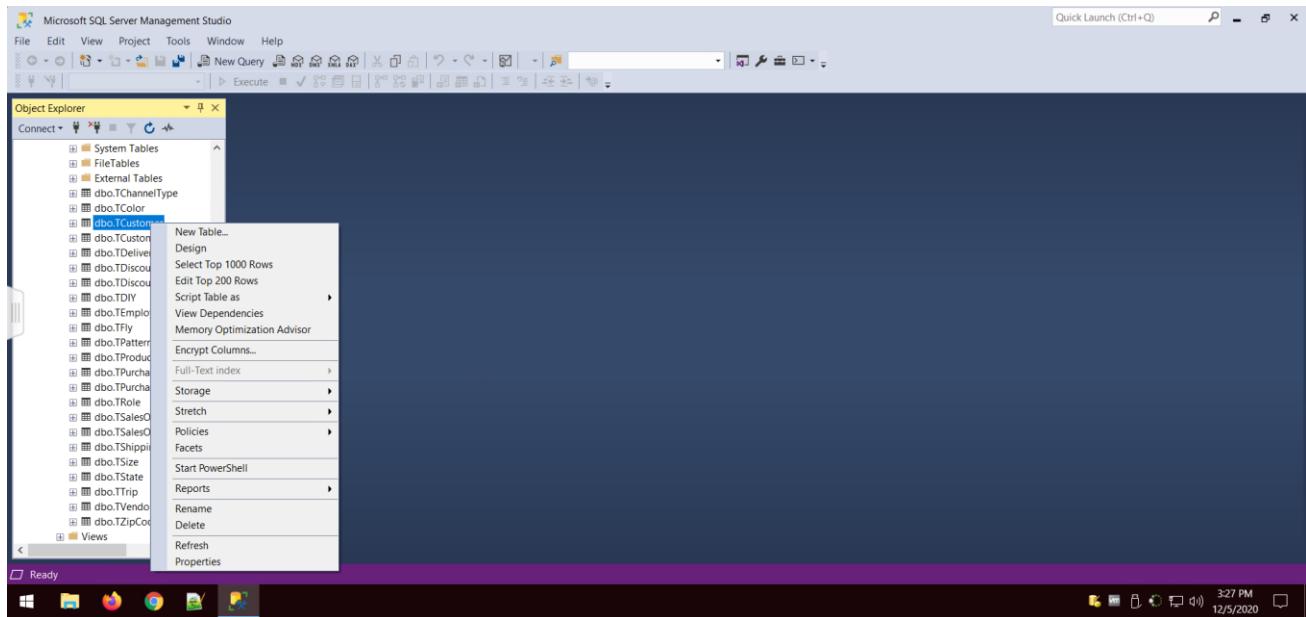
Step 8:

To view the tables that are implemented in this database, click on the + sign next to **Tables** to see the list of tables.



Step 9:

To edit a table, right click on it and choose Edit Top 200 rows.

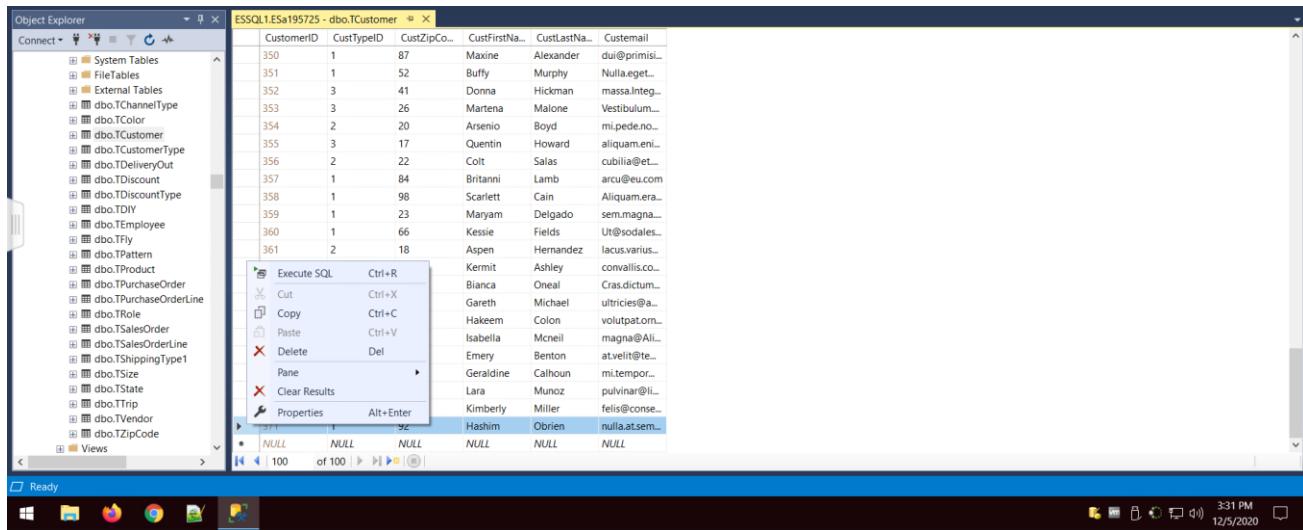


You should be directed to a page like this. Here you can enter data by clicking on the different cells and putting in data as you would in an excel file. If you want a new row, please make sure you enter all the data required for that row. Keep in mind that some data are not allowed to be null.

| CustomerID | CustTypeID | CustZipCode | CustFirstName | CustLastName | CustEmail |
|------------|------------|-------------|---------------|--------------|-----------------|
| 350 | 1 | 87 | Maxine | Alexander | dui@primis... |
| 351 | 1 | 52 | Buffy | Murphy | Nulla.eget... |
| 352 | 3 | 41 | Donna | Hickman | massa.integ... |
| 353 | 3 | 26 | Martena | Malone | Vestibulum... |
| 354 | 2 | 20 | Arsenio | Boyd | mi.pedeno... |
| 355 | 3 | 17 | Quentin | Howard | aliquam.eni... |
| 356 | 2 | 22 | Colt | Salas | cubilia@et... |
| 357 | 1 | 84 | Britannia | Lamb | arcu@eu.com |
| 358 | 1 | 98 | Scarlett | Cain | Aliquam.era... |
| 359 | 1 | 23 | Maryam | Delgado | sem.magna... |
| 360 | 1 | 66 | Kessie | Fields | Ut@sociales... |
| 361 | 2 | 18 | Aspen | Hernandez | iacus.varius... |
| 362 | 3 | 80 | Kermie | Ashley | convallis.co... |
| 363 | 3 | 5 | Bianca | Oneal | Cras.dictum... |
| 364 | 2 | 36 | Garett | Michael | ultricies@a... |
| 365 | 2 | 21 | Hakeem | Colon | volutpat.on... |
| 366 | 1 | 45 | Isabella | Mcneil | magna@Ali... |
| 367 | 3 | 31 | Emery | Benton | at.velit@te... |
| 368 | 2 | 77 | Geraldine | Calhoun | mi.tempor... |
| 369 | 3 | 81 | Lara | Munoz | pulvinar@li... |
| 370 | 1 | 51 | Kimberly | Miller | felis@conse... |
| 371 | 1 | 92 | Hashim | Obrien | nulla.at.sem... |
| * | NULL | NULL | NULL | NULL | NULL |

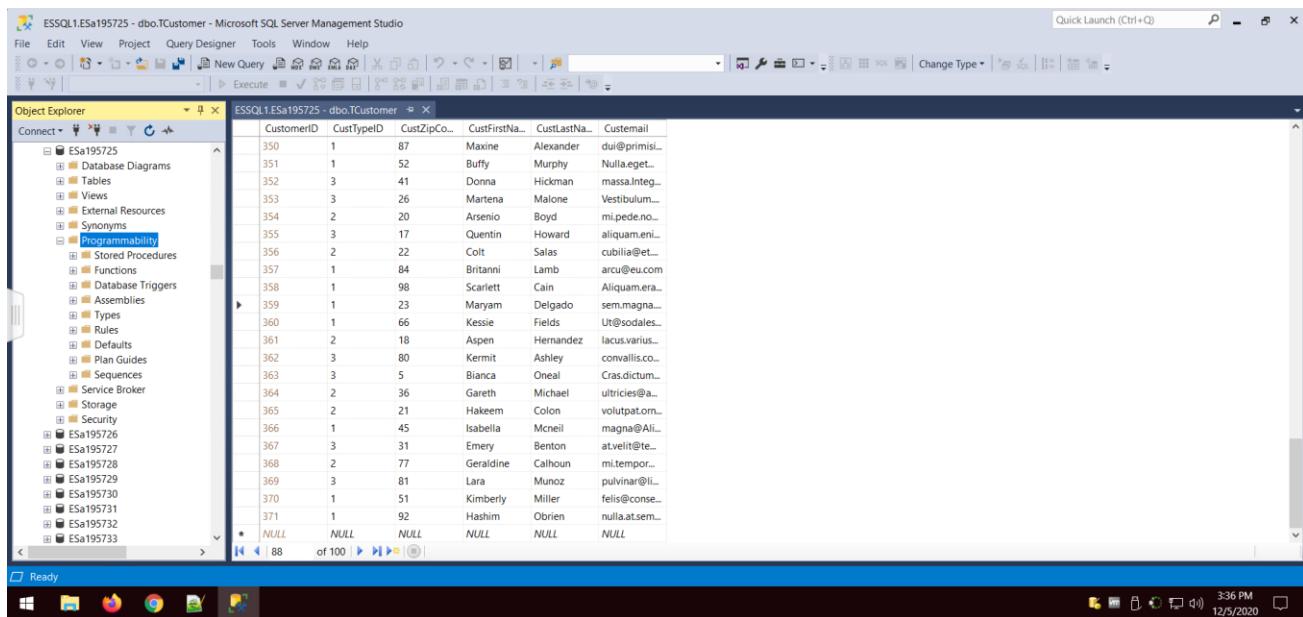
Note that IDs are auto incremented and therefore are **cell read only.

If you made a mistake and wish to delete a cell, just right click on it and choose delete.

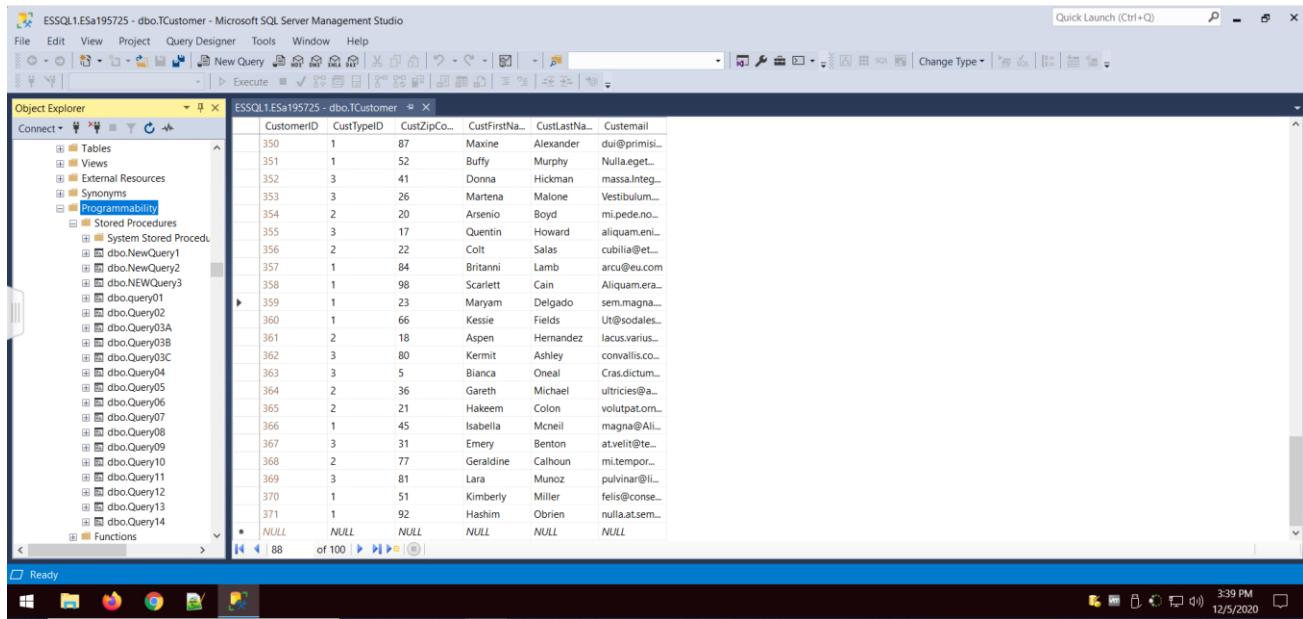


Step 10:

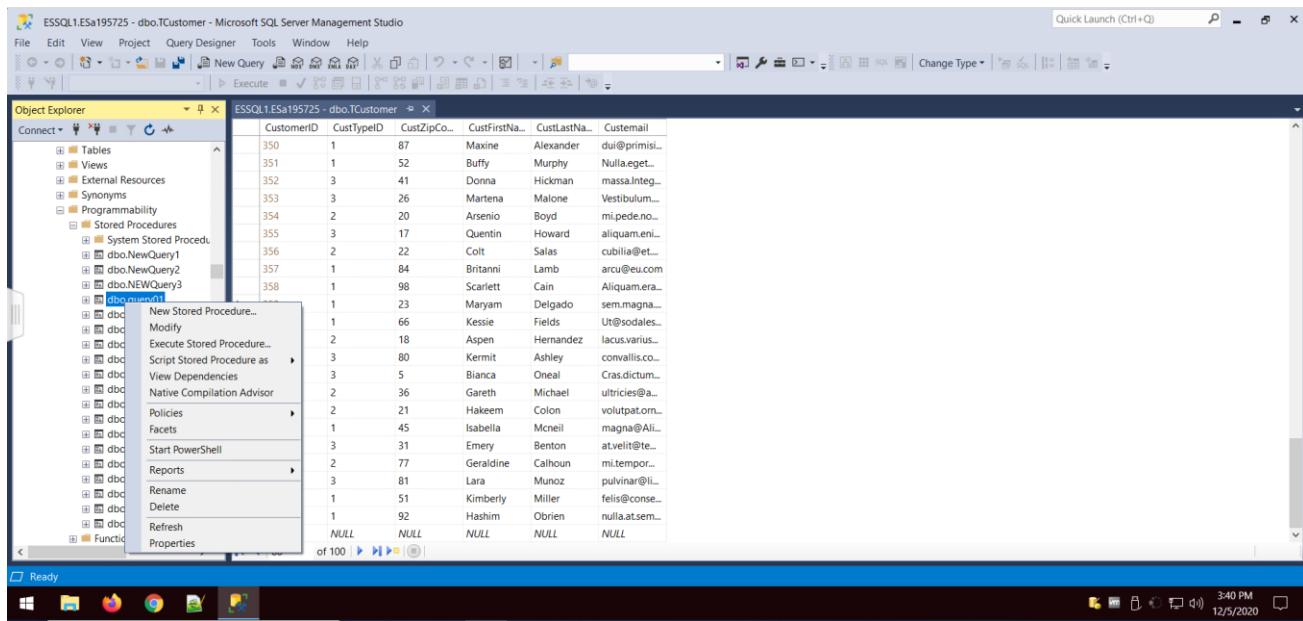
To run queries and get insights on your data, click on the + sign next to **Programmability** to expand it.



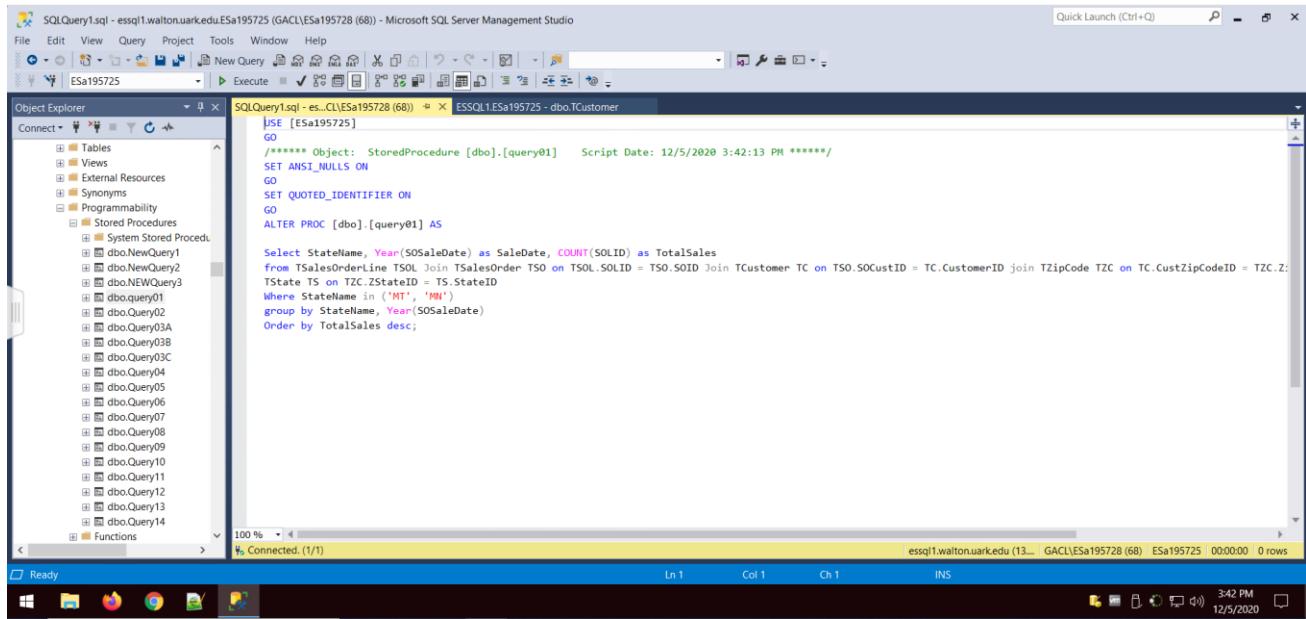
Then click on the + sign next the **Stored Procedures** to expand it. Here you will see a list of the queries that we have already made for based on Elysian Fly's requests.



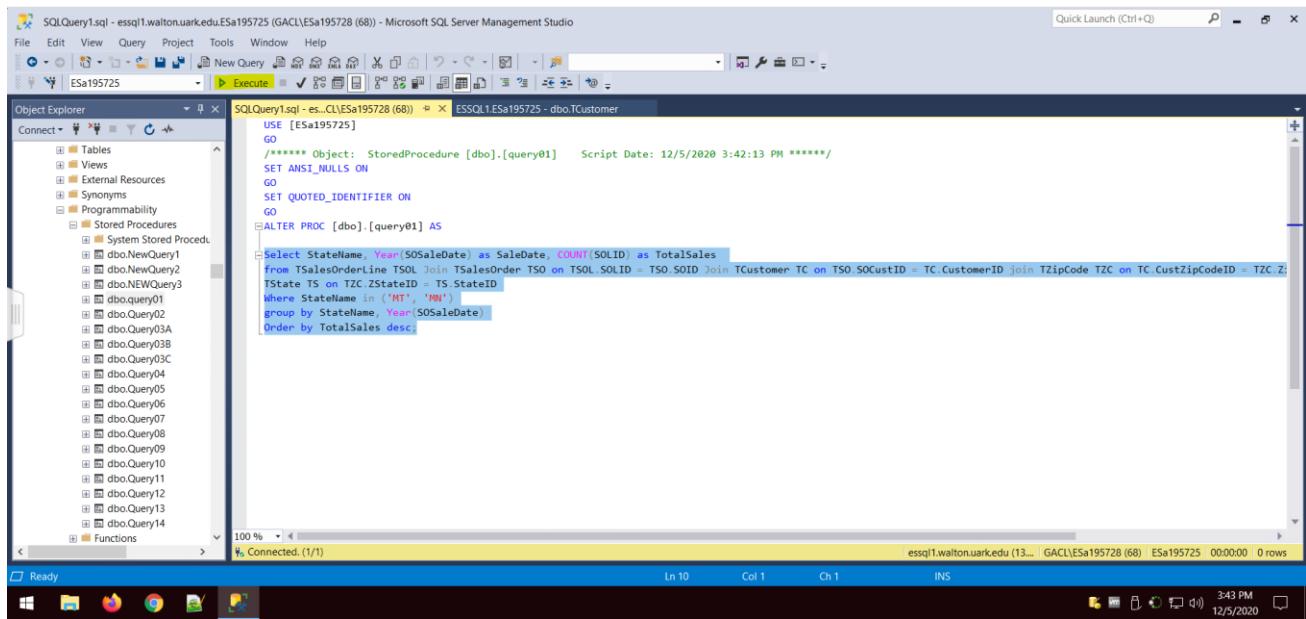
To make it easier for you, you can rename any of the queries. Right click on the query and choose **Rename**.



To run a query, you can choose **Modify** after right clicking on it. You should see a page like this.



It has the code for that query. Select it then click on **Execute** which is highlighted in yellow.



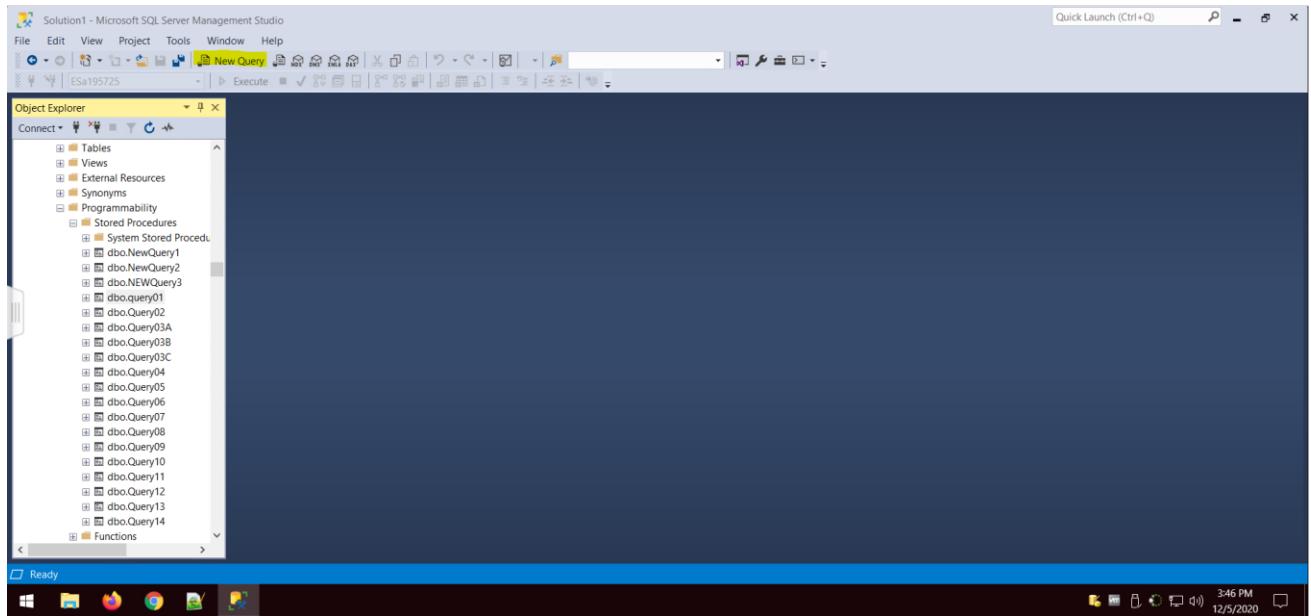
You should see the result of your query.

```

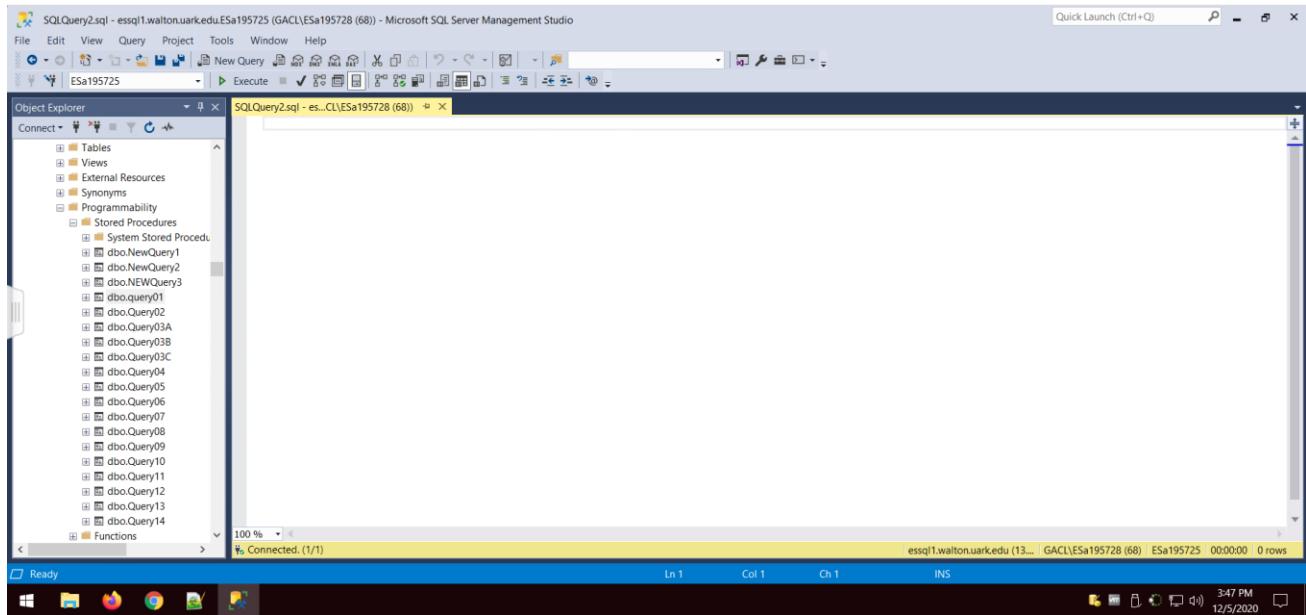
SQLQuery1.sql - esql1.walton.uark.edu\ESa195725 (GACL\ESa195728 (68)) - Microsoft SQL Server Management Studio
File Edit View Query Project Tools Window Help
Execute New Query Quick Launch (Ctrl+Q) x
Object Explorer Connect x
File Object Explorer Connect x
Tables Views External Resources Synonyms Programmability Stored Procedures System Stored Procedures dbo.NewQuery1 dbo.NewQuery2 dbo.query01 dbo.Query02 dbo.Query03A dbo.Query03B dbo.Query03C dbo.Query04 dbo.Query05 dbo.Query06 dbo.Query07 dbo.Query08 dbo.Query09 dbo.Query10 dbo.Query11 dbo.Query12 dbo.Query13 dbo.Query14 Functions
SQLQuery1.sql - es...CL\ESa195728 (68) x ESSQL\ESa195725 - dbo.TCustomer
USE [ESa195725]
GO
***** Object: StoredProcedure [dbo].[query01] Script Date: 12/5/2020 3:42:13 PM *****/
SET ANSI_NULLS ON
GO
SET QUOTED_IDENTIFIER ON
GO
ALTER PROC [dbo].[query01] AS
Select StateName, Year(SOSaleDate) as SaleDate, COUNT(SOLID) as TotalSales
From TSalesOrderLine TSOL Join TSalesOrder TSO on TSOL.SOLID = TSO.SOID Join TCustomer TC on TSO.SOCustID = TC.CustomerID Join TZipCode TZC on TC.CustZipCodeID = TZC.ZIP
TSState TS on TZC.ZStateID = TS.StateID
Where StateName in ('MT', 'MN')
group by StateName, Year(SOSaleDate)
100 % Results Messages
Results
1 MN 2020 2
2 MT 2020 1
3 MT 2019 1
Query executed successfully.
Ln 10 Col 1 Ch 1 INS
essql1.walton.uark.edu (13... | GACL\ESa195728 (68) | ESa195725 | 00:00:00 | 3 rows
12/5/2020 3:45 PM

```

If you wish to make a new query, click on **New query** highlighted in yellow.



A new page will pop up for you to write your code.



If you wish to save your code as a procedure, make sure you include “**Create Proc *NameOfProcedure***” at the top of your code.

What We Learned Throughout This Process

Throughout this process, we had learned many new skills and gained new insight. We have worked together as a team, created a database, and done research for Elysian Fly Company in one semester. While there were challenges and obstacles, we preserved and did the best work that we could. Below is a statement from each team member of Team Mozart on what we all learned throughout this process.

- **Hillary** – Throughout this process, I have learned just how much work goes into creating a database. I have learned how important teamwork is and how necessary it is in a project like this. I have learned that many things in this field build on each other, and it is helpful to understand all topics before moving onto something new, such as the Milestones. I have really enjoyed this project and all of the knowledge that I have gained from it, and I look forward to using this knowledge later on.
- **Lam** – Through this project, I have been able to learn a lot about database construction and interaction. Although the sheer amount of work can be daunting, the learning curve is steep. This project allowed me to connect knowledges and skills from the beginning of the class to the end, from SQL coding to Physical Design. Furthermore, I had a great opportunity to collaborate with other team members, who without doubt, are one of the hardest working people I have ever met.
- **Sara** – Through this process, I have learned how to implement my knowledge about SQL, ERD, Physical Design, and Logical Design in order to create a functioning database. My favorite part from this project has to be creating the logical design and ERD. I really enjoyed working on this hard but rewarding project with my group members and having the ability to increase my understanding and knowledge on the steps and teamwork needed to create a

database for a company.

- **Sahrin** – Based on my experience that I have accumulated through this project; I now understand that building a database is a complicated yet simple task. Factors such as entities and understanding their relationships to one another is what makes this project lengthy and complex. However, if looked at from a broader angle, the task only has three simple steps that must be fulfilled before one can create their database. Through conceptual, logical, and physical design, a developer can note the different factors that play into the relationships of the database and use that information to implement. I personally found the implementation phase the hardest due to it having numerous “mini” tasks within each task. For example, if I would like to create a table- I must also create a data dictionary. Overall, I would say my experience in building my first ever database was incredibly positive. Even though there were rough patches such as my team not being able to find a good time to meet that works for everyone, I managed to learn a lot from the project with their help. Learning the material through books and lecture is one thing, but hands on learning is another!
- **Khady** – Throughout this process I have learned about the work that is done behind databases. Before this project I knew how to retrieve data and always thought that the database that allowed me to do so must be complicated and complex. After doing this project, I know that creating a database is not as hard as it seems. I also got to work on it with other people and with the online setting it was not an easy task. Fortunately, we were able to challenge ourselves and come up with a database.
- **Team** – What we learned about throughout this project for Elysian Fly Company was the importance of efficient data and the importance of queries in executing specific functions in a database. We learned about what is necessary to make effective queries that display data to Elysian Fly Company. We also learned how to effectively use SQL Server; like creating tables and queries, and to deal with all the complexities of SQL Server. What we realized through this project was how SQL, ERD, Logical Design, and Physical Design are connected and all necessary parts in creating an effective database that our customer, Elysian Fly Company can implement in order to improve their business and collect specific information they need on areas within their company. We also learned how important is it to constantly update and revisit old milestones to improve our presentation and be able to effectively display all the information we have learned throughout this project. We were surprised about how much information can be stored in a database and how challenging creating effective and necessary queries are in accomplishing the mission of Elysian Fly Company.
- **Project Management** – Using the project management tool, we were able to track one another’s progress to make sure that everyone was participating. We used it to list each team members contributions, so that everyone would get the correct amount of credit at the end of the project. The project management toll helped to keep all team members accountable and will be a useful tool for future projects.

Appendix

Team Contract

Team Mozart is a database design and development firm that offers a wide range of IT services to provide effective data management and storage. For more than 16 years, Mozart has been delivering top-notch curations of tailor-made databases and consultation for unique businesses all over the global community.

Team E - MOZART

Above and beyond



| Team Member | Email | Phone | Strengths | Availability to Meet |
|----------------|---------------------|----------------|-----------------------|-----------------------------------|
| Lam Nguyen | Lamnguyen99@ou.edu | (405) 361-6065 | Hard-working | 24/7 via Microsoft Teams and Zoom |
| Sara Mullins | sara.mullins@ou.edu | (816) 269-3743 | Positive | 24/7 before 1:00 pm |
| Hillary Neaves | hillaryn23@ou.edu | (405) 397-2850 | Organized and Helpful | 24/7 between 10:00am and 1:00pm |

| | | | | |
|----------------|----------------------|------------------|-------------------------------------|------------------------------|
| Khady Ndoye | Khady.Ndoye-1@ou.edu | (405) 795 - 6531 | organized | 24/7 via email |
| Sahrin Moytree | moytree@ou.edu | (405) 371-5243 | Reliable, organized, and determined | 24/7 via advanced scheduling |

Team E, Mozart, explicitly hire individuals who are creative, hard-working, team players, organized, and most importantly harbor a positive outlook. The team member must display admirable qualities that complement the company's end goal.

The expectations for each team member must be met through the individual's work ethic. He or she must participate in every task allotted, must be willing to cooperate and ease the burden of tasks on other team members if needed, and must complete honest team evaluations as the end of the semester for each member

Data Dictionary Model

| Table | PK? | Field Name | Data Type | Size | Null | Default | References | Sample |
|-----------------|-----|-----------------------|------------|--------|----------|---------|---------------|------------------|
| TState | Yes | StateID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | StateName | Varchar | 2 | Not Null | 1 | | OK |
| TZipCode | Yes | ZipCodeID | Int (Auto) | 5 | Not Null | 1 | | 1 |
| | No | ZipCode | Varchar | 10 | Not Null | | | 73072 |
| | No | ZCity | Varchar | 50 | Not Null | | | Norman |
| | No | ZStateID | Int | | Not Null | | TState | 1 |
| TCustomerType | Yes | CustTypeID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | CTDescription | Varchar | 100 | Not Null | | | One-Off Tourist |
| TDiscountType | Yes | DiscountTypeID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | DTAmount | Decimal | (3,2) | Not Null | | | 0.05 (10%) |
| TDiscoun | Yes | DiscountID | Int (Auto) | 1 | | | | 1 |
| | No | DDiscounTypeID | Int | | Not Null | | TDiscounType | 1 |
| | No | DMoneySaved | Decimal | (10,2) | | | | 1200.50 |
| TSalesOrderLine | Yes | SOLID | Int (Auto) | 1 | Not Null | | | 1 |
| | No | SOLProductID | Int | | Not Null | | TProduct | 1 |
| | No | SOLSOID | Int | | Not Null | | TSalesOrder | 1 |
| | No | SOLQuantity | Int | | Not Null | | | 1 |
| | No | SOLStatus | Varchar | 50 | Not Null | | | 1 |
| TColor | Yes | ColorID | Int (Auto) | 1 | | | | 1 |
| | No | ColorDescription | Varchar | 50 | Not Null | | | Red |
| TTrip | Yes | TripProductID | Int (Auto) | 1 | | | | 1 |
| | No | ProductID | Int | | Not Null | | TProduct | 1 |
| | No | TripCustID | Int | | Not Null | | TCustomer | 1 |
| | No | TripGuideEmpID | Int | | Not Null | | TEmployee | 1 |
| | No | TripIsFishingSeason | Varchar | 3 | Not Null | | | 1 |
| | No | TripDate | Date | | Not Null | | | January 25, 2020 |
| | No | TripZipCode | Int | | Not Null | | TZipCode | 1 |
| | No | TripWeather | Varchar | 100 | Not Null | | | Sunny |
| | No | TripIsBooked | Varchar | 50 | Not Null | | | Booked |
| | No | TripNumOfDays | Int | | Not Null | | | Five |
| | No | TripNumOfParticipants | Int | | Not Null | | | Eleven |
| | No | TripRegulations | Varchar | 100 | Not Null | | | |
| | No | TripGuidePreferences | Varchar | 100 | Not Null | | | Guide One |
| TProduct | Yes | ProductID | Int (Auto) | 1 | | | | 1 |
| | No | ProdName | Varchar | 50 | Not Null | | | Fly |
| | No | SalePrice | Int | 1 | Not Null | | | 50.00 |
| | No | ProductType | Varchar | 50 | Not Null | | | DIY |
| TCustomer | Yes | CustomerID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | CustTypeID | Int (Auto) | | Not Null | 1 | TCustomerType | 1 |

| | | | | | | | | |
|---------------------|-----|--------------------|---------------|-------|-----------------|---|--------------|-------------------|
| | No | CustZipCodeID | Int (Auto) | | Not Null | 1 | TZipCode | 1 |
| | No | CustFirstName | Varchar | 50 | Not Null | | | John |
| | No | CustLastName | Varchar | 50 | Not Null | | | Karinsky |
| | No | Custemail | Varchar | 50 | Not Null | | | John.Karinsky.com |
| TRole | Yes | RoleID | int | | Not Null | 1 | | 1 |
| | No | RFunction | Varchar | 100 | Not Null | | | Sales |
| | No | RDescription | Varchar | 100 | Not Null | | | Manage Vendor |
| TDIY | Yes | DIYBundleID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | DIYFlyProductID | Int | | Not Null | | TFly | 2 |
| | No | DIYFlyProduct | Int | | Not Null | | TFly | 3 |
| | No | DIYDescription | Varchar | 100 | Not Null | | | Family Packages |
| | No | DIYQuantity | Int | | Not Null | | | 6 |
| TFly | Yes | FlyProductID | Int(Auto) | | Not Null | | | 1 |
| | No | FlyName | Varchar | 10 | Not Null | | | FlyA |
| | No | FlyColorID | Int | | Not Null | | TColor | Blue |
| | No | FlySizeID | Int | | Not Null | | TSize | 5 |
| | No | FlyPatternID | Int | | Not Null | | TPattern | Stripped |
| | No | FlyIsDurable | Varchar | 3 | Not Null | | | Yes |
| | No | FlyIsGoodQuality | Varchar | 3 | Not Null | | | Yes |
| | No | FlyIsEffective | Varchar | 3 | Not Null | | | Yes |
| | No | FlyIsTestPassed | Varchar | 3 | Not Null | | | Yes |
| | No | FlySalesPrice | Decimal | (3,2) | Not Null | | | 3.99 |
| | No | FlyPurchaseOrder | Decimal | (3,2) | Not Null | | | 4.99 |
| | No | FlyQty_OH | Int | | Not Null | | | 1 |
| | No | FlyQty_Commanded | Int | | Not Null | | | 1 |
| | No | FlyQty_Avail | Int | | Not Null | | | 1 |
| | No | FlyQty_BackOrdered | Int | | Not Null | | | 1 |
| | No | FlyLocation | Varchar | 10 | Not Null | | | BackStock |
| TChannelType | Yes | ChannelTypeID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | ChanDescription | Varchar | 100 | Not Null | - | - | Online |
| | No | ChanHandler | Varchar | 100 | Not Null | - | - | Salesperson Name |
| TSalesOrder | Yes | SOID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | SOCustID | Int | - | Not Null | - | TCustomer | 1 |
| | No | SOEmpID | Int | - | Null Allowed | - | TEmployee | 1 |
| | No | SOPackTypeID | Int | - | Not Null | - | TPack | 1 |
| | No | SOChannelTypeID | Int | - | Not Null | - | TChannelType | 1 |
| | No | SODiscountID | Int | - | Not Null | - | TDiscount | 1 |
| | No | SOItemName | Varchar | 100 | Not Null | - | - | Fly |
| | No | SOPaymentMethod | Varchar | 50 | Not Null | - | - | Cash or Card |
| | No | SOSaleDate | Varchar | 100 | Not Null | - | - | 01-25-20 |
| | No | SOStatus | Varchar | 50 | Not Null | - | - | Delivered |
| TProduct | Yes | ProductID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | ProdName | Varchar | 100 | Not Null | - | - | Fly |
| | No | SalePrice | Decimal | 50 | Not Null | - | - | 20.00 |
| | No | ProductType | Varchar | 100 | Not Null | - | - | Fly or Pack |

| | | | | | | | | |
|---------------------------|-----|---------------------|---------------|--------|----------|---|----------------|---------------|
| TPattern | Yes | PatternID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | PatDescription | Varchar | 100 | Not Null | - | - | Stripe |
| | No | PatColor | Varchar | 100 | Not Null | - | - | Red |
| | No | PatSize | Varchar | 100 | Not Null | - | - | Small |
| TSize | Yes | SizeID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | SizeDescription | Varchar | 100 | Not Null | - | - | Medium |
| TShippingType1 | Yes | ShippingTypeID1 | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | SPTDescription1 | Varchar | 100 | Not Null | 1 | - | Ground |
| TDeliveryOut | Yes | DeliID | Int (Auto) | - | Not Null | 1 | - | 1 |
| | No | DOSOID | Int | 100 | Not Null | - | TSalesOrder | 1 |
| | No | DOShipDate | Date | 100 | Not Null | - | | 11/13/2020 |
| | No | DOShippingCost | Float | (10,2) | Not Null | - | | 123.99 |
| | No | DOShippingTypeID | Int | - | Not Null | 1 | TShippingType1 | 1 |
| | No | DOCARRIER | Varchar | 100 | Not Null | - | | FedEx |
| | No | DOTrackingNumber | Varchar | 100 | Not Null | 1 | | 123456789 |
| TPurchaseOrder | Yes | TPurchaseOrderID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | POEmpID | Int | | Not Null | | TEmployee | 2 |
| | No | PODate | Date | | Not Null | | | 2019-07-13 |
| | No | POTotal | Int | | Not Null | | | 2500 |
| TPurchaseOrderLine | Yes | PurchaseOrderLineID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | POLPOID | Int | | Not Null | | TPurchaseOrder | 1 |
| | No | POLDFlyID | Int | | Not Null | | TFly | 1 |
| | No | POLQuantity | Int | | Not Null | | | 6 |
| | No | POLLineTotal | Int | | Not Null | | | 143 |
| TVendor | Yes | VendorID | Int (Auto) | | Not Null | 1 | | 1 |
| | No | VName | Varchar | 50 | Not Null | | | Diam Corp |
| | No | VVendorAccount | Int | | Not Null | | | 128 |
| | No | VTerms | Varchar | 100 | Not Null | | | 2/10 net 30 |
| | No | VZipCodeID | Int | | Not Null | | TZipCode | 2 |
| | No | VTimesUsed | Int | | Not Null | | | 5 |
| | No | VDollarsSpent | Int | | Not Null | | | 15000 |
| | No | VPhone | Varchar | 15 | Not Null | | | (393)930-6609 |
| | No | VContactName | Varchar | 50 | Not Null | | | Kyle Carney |
| | No | VFlyingTechniques | Varchar | 100 | Not Null | | | Triple Top |

Project Management

| Team Members: Sahrin Moytree, Sara Mullins, Khady Ndoye, Hillary Neaves, Lam Nguyen | | | | | | | |
|--|---|------------------|------------|-------------------|----------------|--------------------|------------------|
| Project Start Date | 10/8/2020 | Project End Date | 12/06/2020 | Cost (per 60 min) | \$300 | | |
| Elysian Fly Company | Student Name | Duration (Min) | % Complete | Planned Minutes | Actual Minutes | Difference Minutes | Subtotal Minutes |
| Milestone 1 | | | | | | | |
| Read Case + Prepare Questions for client | All Team Members | 60 | 100% | 120 | 60 | 60 | |
| Client Meeting | All Team Members | 15 | 100% | 15 | 15 | 0 | |
| ERD Design | Khady Ndoye, Hillary Neaves, Lam Nguyen, Sara Mullins | 60 | 100% | 120 | 60 | 60 | |
| Assumptions | All Team Members | 60 | 100% | 120 | 60 | 60 | |
| Write-up preparation | Sahrin Moytree | 240 | 100% | 120 | 240 | -120 | |
| Sub Total | | | | | 60 | 435 | \$2,175 |
| Milestone 2 | | | | | | | |
| Feedback Corrections | Hillary Neaves | 120 | 100% | 60 | 120 | -60 | |
| Normalization | Khady Ndoye, Sara Mullins, Lam Nguyen | 240 | 100% | 60 | 240 | -180 | |
| Write-up preparation | Sahrin Moytree | 120 | 100% | 60 | 120 | -60 | |
| Reviewing and Finalizing | Hillary Neaves | 60 | 100% | 60 | 60 | 0 | |
| Sub Total | | | | | -300 | 540 | \$2,700 |
| Milestone 3 | | | | | | | |
| Milestone 3 Document | Hillary Neaves | 120 | 100% | 60 | 120 | 0 | |
| Presentation | All Team Members | 90 | 100% | 90 | 90 | 0 | |
| TState, TZipCode, TCustomerType, TDiscountType, TDiscount, TSalesOrderLine, TColor, TRip, TFly | Khady Ndoye | 240 | 100% | 120 | 240 | 120 | |
| TChannelType, TSalesOrder, TProduct, TSize, TPattern | Hillary Neaves | 240 | 100% | 120 | 240 | 120 | |
| TBOW, TBOWKnown, TRole, TCustomer | Lam Nguyen | 240 | 100% | 120 | 240 | 120 | |
| TDIY, TEmployee | Sara Mullins | 60 | 100% | 120 | 60 | -60 | |
| | Sahrin Moytree | | 100% | 120 | 0 | -120 | |
| Query 1 | Khady Ndoye | 60 | 100% | 60 | 60 | 0 | |
| Query 6 & New query 1 | Lam Nguyen | 60 | 100% | 60 | 60 | 0 | |
| New queries 2 & 3 | Hillary Neaves | 60 | 100% | 60 | 60 | 0 | |
| No queries | Sara Mullins | | 100% | | | | |
| No queries | Sahrin Moytree | | 100% | | | | |
| Reviewing and Finalizing | Khady Ndoye | 60 | 100% | 60 | 60 | 0 | |
| Sub Total | | | | | 0 | 1230 | \$6,150 |
| Final Submission | | | | | | | |
| Milestone 3 corrections | All team members | 60 | 100% | 60 | 60 | 0 | |
| Document updates | Sahrin Moytree, Sara Mullins, Hillary Neaves, Khady Ndoye | 240 | 100% | 240 | 240 | 0 | |
| No tables | Khady Ndoye | | | | | | |
| No tables | Hillary Neaves | | | | | | |
| TPurchaseOrder, TPurchaseOrderLine, TVendor, TFly | Lam Nguyen | 60 | 100% | 60 | 60 | 0 | |
| | Sara Mullins | | | | | | |
| TShippingType, TDeliveryOut | Sahrin Moytree | 60 | 100% | 60 | 60 | 0 | |
| Queries 1 to 7 | Khady Ndoye | 240 | 100% | 240 | 240 | 0 | |
| No queries | Hillary Neaves | | | | | | |
| No queries | Lam Nguyen | | | | | | |
| Queries 8 to 14 | Sara Mullins | 240 | 100% | 240 | 240 | 0 | |
| No queries | Sahrin Moytree | | | | | | |
| User Documentation | Khady Ndoye | 60 | 100% | 60 | 60 | 0 | |
| Reviewing and Finalizing | All team members | 60 | 100% | 60 | 60 | 0 | |
| Sub Total | | | | | 0 | 1020 | \$5100 |
| | | | | | Total | 3,065 | \$16,125 |