**Goals**

Understanding of the structure of star-schema and concept of data cubes, differences of star schema from a normalized database schema, and how to create an example star-schema in Excel using PowerPivot along with the use of slicers in Excel pivot tables for slicing and dicing data cubes. Additionally, the assignment aims to provide developing skills in using Tableau Prep, Alteryx and Python as ETL tools, and SQL to interact, extract, transform and load data from a relational database into a data warehouse.

You may partner another classmate. I can allow one group of three to accommodate odd number of students in the class.

**Objectives**

After completion of this assignment students should be able to:

* List at least one difference between a star and a normalized schema
* Articulate the differences between measures and dimensions in data
* Identify slow changing dimensions
* Design and draw a star schema structure by inspecting different data sources (mainly tables) and structures
* Develop SQL queries to address strategic, operational, and analytical data requests
* Insert data into a data warehouse with Alteryx, Python and Tableau Prep
* Create various types of dashboards (strategic, operational, and analytical) in Tableau and/or PowerBI using the data from the dimensional models

**Preparatory Work**

* Go through [the pdf document](https://lmu.box.com/v/BSAN-6040-Datawarehouse) that explains the concepts of Data Warehouse and shows how to implement them using Talend for the sakila database. We will not use Talend, but the concepts are very similar. The document is 168 pages long, but most of them are pictures. You can skip the first few pages that describes logging into O’Riley and start with Data Warehousing on page 12.
* You have to have your Tableau and Tableau Prep installed and running with ability to connect to MySQL servers.
* You have to make sure that Alteryx is installed and running in your VDI.
* You also have to have your PowerBI set up in your VDI. We will cover it in class.
* Make sure that you know how to crate and use databases in both LMU Build and AWS RDS services. Collect all the connection information before you begin.

**What we are trying to achieve**

This assignment (and your project) is a way for you to understand the modern data architecture needed for handling large datasets and extracting information in an efficient fashion for effective decision making. The RDBMS worked for quite some time but with the growth of data and needs for the data for decision making we needed new ways to handle data because:

* RDBMS are great for writing structured data and for keeping detailed records but
* The Joins across the tables are expensive and for large dataset queries can take a long time to run
* Visualization tools run really slow when you connect them directly to a RDBMS with large number of records and query the information for the visualization
* RDBMS cannot handle different data types
* Large data cannot be easily read with traditional tools such as
  + Excel
  + Text Editors
* Creating tables from large files (with millions of records) and then loading those tables to RDBMS are very difficult and resource intensive
* RDBMS are not adept in updating the meta data when new rows are added. Indices (or keys) are to be recreated and that can take time.

To overcome the above limitations, new architecture such as Data Warehouse and Data Lakes are created that can handle large amount of data. In the case of Data Lake:

* The raw data can just be in csv format in a storage (no need to extract the records, divide them in tables, etc.)
* If you need data from multiple sources and they are in the same storage area, Data Lake programs (like Athena) allows you to connect them and retrieve information (they have to have connecting fields just like we need in JOIN)
* Data often are loaded and then partitioned in columnar database (e.g. Parquet) format that increases the speed and efficiency of data scan tremendously. As a result, visualizations created by using queries on Data Lakes respond very quickly.
* In cloud, you can set simple triggers to update the metadata when new data is added.

However, Data Lake still has its limitations because it cannot support pre-created schema. It offers flexibility but not that much of structure. Business decisions are often repetitive and rely on aggregated data. That is where Data Warehouse comes in because it can:

* Create very efficient structure for quickly slicing and dicing data through dimensional models,
* Can store aggregated information that are often used by businesses at an appropriate level of details (called grains),
* Are structured through schemas (like the RDBMS but different that 3NF) such as star schema and therefore, specialized hardware optimized for handing massive amount of data can be used for faster performance,
* Provides a natural hierarchical organization of information (because of the schema).

***In this assignment, the end result will be a Data Warehouse that Tableau/Power BI can connect to and create visualizations. We will start with the following types of data sources:***

1. ***CSV file (mimicking the flat file on a computer or storage)***
2. ***RDBMS with a few tables (mimicking a corporate transactional database) that can come from multiple servers (e.g. LMU Build and AWS)***

***The end result will be a visualization through Tableau and/or Power BI on the data from Data Warehouse created in your LMU Build or AWS RDS container. The Data Warehouse will be in star schema with fact and dimension tables.***

**Why so many input sources?**

In real life scenario, you will have multiple data sources in multiple locations and servers and you have to consolidate them into one area (your DW). I want you to see how that is done by using ETL tools. Hence the multiple sources.

You will use Alteryx, Python and Tableau Prep as the ETL tool and use the LMU Build or AWS RDS container as your RDBMS. Conceptually, the overall flow of the project will be as shown below:

**TASKS**

1. **Initial Data Exploration.**

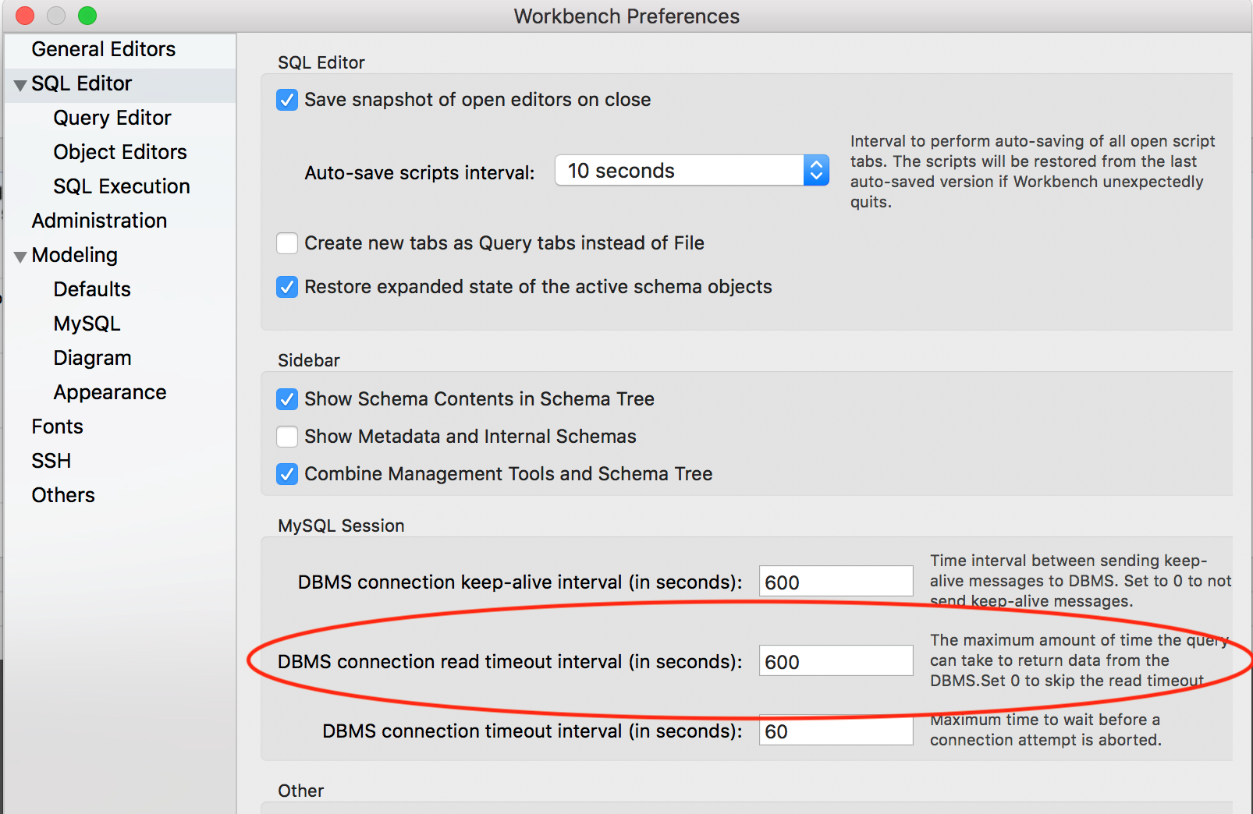
Download the files from the **Assignment 2 files** link on Brightspace. Explore each of the files by extracting some sample data in the command prompt. **Note that these are large files with millions of records and thus it is not prudent to double-click on then or open them in Excel**. Here is a link that tells you how to sample data from real large csv files. <https://www.csvexplorer.com/blog/open-big-csv/> To get to Windows PowerShell, just search for it in your computer and then launch it.

I am also assuming that your MySQL Workbench is all set up.

You can refer to the Dillard’s Relational Database Entity Relationship Diagram at the end of this document.

If you run into this error when running a query anytime during the completion of this assignment, then, of course, let me know but updating the timeout under Preferences from 30 seconds to 600 seconds can help too.

“Error Code: 2013. Lost connection to MySQL server during query”



1. **Loading the files in your AWS and LMU RDS database.**

**Purpose of this step: Understanding ETL and doing some simple task with Alteryx and Tableau Prep. Minimal data transformation is needed.**

Since we would be using the records in the csv files that represent that tables in the relational DB diagram for Dillard’s, we have to upload them in a relational database. That would facilitate quick access to the data and a lot more flexibility. You will be using Alteryx and Tableau Prep to accomplish them. Please see below for the detailed instructions.

1. Set up the Relational Database in LMU Build according to the ERD shown at the end. The DW model will be in the LMU Build or AWS RDS database. You have to create new schemas for the assignment. Create a schema/database named ***dillards\_relational\_model\_lmu*** in LMU Build and schema as ***dillards\_dim\_model\_aws*** in AWS (or ***dillards\_dim\_model\_build*** if you are using LMU Build***)***

Steps for LMU Build.

1. Log into your LMU Build account
2. Go to Database area and create a database named <your lmu build id>\_dillards\_relational\_model\_lmu
3. Add one of the existing users to the database (make sure you remember the password) and give it all privileges
4. Save and then connect to this database using MySQL Workbench to make sure that it is working, and you can see the database.

Steps for AWS.

1. Go to your AWS account through MySQL Workbench
2. Open a Query tab and write CREATE SCHEMA dillards\_dim\_model\_aws;
3. Refresh the schema views on the left hand side to make sure that the schema/database shows up.

**Note: Use the proper convention for your sql table and field names that you will create in your databases. I will also strongly suggest to create smaller versions of each of the dataset to test out the loadings and the overall design before you start loading millions of rows of data only to find that it fails at the end. That will be time consuming and frustrating. You can get the smaller versions of the data by going to my LMU Build account and then downloading the tables that starts with a2. You will see that each of them corresponds to an entity on the ERD provided at the end.**

**Loading the Data**

*Tableau Prep*

Use Tableau prep to load the **stores.csv and departments.csv** to your LMU Build database.

The overall flow should look like the following:

Diagram

Description automatically generated with low confidence

Steps:

1. Create a **stores** table first in your LMU database with the correct data types as shown in the relational schema. Use the lower case naming conventions. The four fields are store\_id, city, state and zip. Make sure that zip and city are in VARCHAR. Use CREATE table command.
2. Open Tableau Prep ([here is a link to Tableau Prep tutorials](https://help.tableau.com/current/prep/en-us/prep_welcome.htm) and link to [video](https://www.tableau.com/learn/tutorials/on-demand/getting-started-tableau-prep?playlist=509088)) and connect to the **stores.csv** file as a data input.
3. Insert a cleaning step and clean the data. You have to rename the field names to the same one as in your database table. Make sure that all the fields are of correct types. Trim the spaces from all the string fields.
4. Add an output step and connect that output step to your LMU Build database for this assignment and then select the stores tables from the database. You can then decide if you want to replace the table or append the data.
5. Make sure that the mapping of the fields is correct. If some fields are not mapped, then you have to go back to your cleaning step and name the fields same as the ones in the stores table in the database.
6. You can use the Customer SQL tab to alter the table before and after by using SQL. For example, you can drop a primary key or a field in the table, alter the nature of the field before the flow and add a primary key and/or change data type of a field, etc. after the flow is run. Multiple alter activities can be specified with a comma. For example, if I wanted to change the store\_id to an INT and make it a primary key AFTER the flow is run, I will write the following:
   1. ALTER TABLE stores
   2. MODIFY COLUMN store\_id INT,
   3. ADD PRIMARY KEY(store\_id);
7. Run the flow and prep should load the content of the stores.csv to the stores table in your lmu database.
8. Using Tableau Prep, also load the content of the departments.csv file to a table in your LMU Build database.
9. If you do all of the above steps correctly, your flow should look like the following

*Python/Alteryx*

***Why Python? Tableau Prep is slow. Python is industrial level tool and is really fast in loading data. That is why we would like to use it for loading the large files like transactions, skus, etc. You can also test Alteryx in loading large dataset with millions of records to see how it performs and can then decide if it would be an appropriate tool for that.***

Use Python/Alteryx to load the following files to your LMU Build database:

1. **transactions.csv (1 million rows).**  You can create the output table on the fly. Once the table is loaded, check all of the fields to make sure that they are same as the ones shown in the relational schema. Note that strings or long CHAR should be VARCHAR in the MySQL database while short CHAR should be character. Note that you need a delimited file as input (typically csv), Dask/Pandas and a database as output through a database connection.
2. **skus.csv (about 1.5 million rows).** You can create the table on the fly or can create the empty table in LMU Build first and then map the fields with the fields in the csv file.
3. Create the empty sku\_stores table in your LMU database and then using Python or Alteryx, load the data from the **sku\_stores.csv** file to this table.

Create a dim\_date table in Excel using the date ranges in the transactions data and the various hierarchy fields (week, month, quarter, year, etc.). Please watch [this video](https://lmu.box.com/s/63c8zo2j3pjf64bsubpltco1wcopdmuo) to get an idea of the structure of the dim\_date table as well as how to create it.

Load the dim\_date table to your aws database using either Tableau Prep or Alteryx.

1. **Data Warehouse Purpose**

The next step is to build a dimensional model and a data warehouse using the data from Dillard’s. To be able to do that, you have to identify the facts and dimensions which are dependent on the information needs of a decision-maker. For this assignment, you are the decision-maker. What do you think will be some of the information that you would need to find from the data warehouse at the Strategic, Operational, and Analytical level? Some examples of typical data requests are given below. Use Chat GPT to get more ideas about such data requests. Feed Chat GPT the names of the tables and also any other details that you want to explore. Note that those data requests will determine the tables and columns you will create for your data warehouse, as well as the lowest grains of the DW design, so you have to really understand them.

**Strategic**

1. Total revenue per month for April, May, and June in 2005.

2. Total purchase count per month for April, May, and June in 2005. Return the month name and the month's total purchase count.

3. Total profit per year and also the percentage profit that came for each of the months in 2004 and 2005. Return the year, month name, month's total profit and the percentage based on the annual profit for each year.

4. Top 5 brands that are returned most frequently and the associated lost revenue.

**Analytical:**

1. Top 10 SKUs based on quantity sold for May 7, 2005 to May 14, 2005. Return the SKU and the quantity sold for the SKU.

2. Top 3 department and city combinations based on revenue for December 1, 2004 to December 31, 2004. Return the department and city and the revenue for the department and city combination.

3. The number of returned items (STYPE = 'R') for each day of the week for June 2005. Return the day of the week name, i.e. Monday, Thursday, etc. and its returned items total.

4. The sales trend for each of the months in 2005. This will allow to see the trends for each month and identify potential opportunities for promotions.

**Operational:**

1. Average revenue per transaction from April 1, 2005 to April 30, 2005 for stores in Texas. Return the date and the average revenue per transaction for the date.

* To get average revenue per transaction (column names would change based on the schema you created) use:

(SUM(purchase\_revenue) / SUM(purchase\_transaction\_count)) AS average\_revenue\_per\_transaction

2. Daily purchase count for a given department from April 7, 2005 to April 14, 2005. Return the date and the date's purchase count. You can test the query with any of the departments.

3. The 5 lowest performing stores for April 1, 2005 to April 30, 2005 based on purchase revenue. Return the store ID and the store's total revenue for the entire date range.

4. **Data Warehouse Design**

Before you create the data warehouse, identify the following based on the data requests required in part 3. Make a copy of [this spreadsheet](https://lmu.box.com/v/BSAN-6060-star-schema-Excel) to plan your design. Save the file as ***star\_schema\_details.xlsx***

Step 1. Specify the fact tables. For this assignment, you will need 2 fact tables (one for purchased items and another for the returned items). You can do it with one fact table as well, but you have to then use a CASE for calculating the profit to ensure that returned items show zero profit. You have to specify,

1. Table names
   * 1. Should have a "fact\_" prefix (in practice suffix is preferred but since we started with the prefix convention for our practice work, let us stick to that).
   1. Field names
   2. Mark with a Y next to the field name
      1. Primary key?
      2. Foreign key?
      3. Measure?
      4. Calculated?
   3. Aggregate Function to be used - SUM, COUNT, AVERAGE, etc.
   4. The grain level should be set to date (daily) and other dimensions decided by your design.

Step 2. Specify the dimension tables. For this assignment, you will need 3 dimension tables, which includes the dim\_date table. For each table, specify:

1. Table name
   * 1. Should have a "dim\_" prefix
   1. Field Name
      1. List the fields in the order of the hierarchy if one exists. Start from high to low.
   2. Hierarchy? - Mark Y next to the field name if it's part of a hierarchy
   3. Primary key? - Mark with a Y next to the field name

Step 3. Draw a Star Schema with the 3 dimension tables and either one or two fact tables identified from the steps above. You can use any of the drawing tools available to you such as Draw.io, Visio, Google Drawings, [Lucidchart](https://www.lucidchart.com/pages/templates/er-diagram/database-er-diagram-template), etc. Save the file as ***data\_warehouse\_erd.jpg or data\_warehouse\_erd.png***

5. **Data Warehouse Implementation**

Create the fact tables and the dimensions tables in your AWS MySQL RDS instance. Please note that by default, date is a dimension and thus must be added to the entire schema as its own dimension table. The transaction date is used as a way to link the date dimension with the fact tables.

Your final star schema tables will reside in AWS. You have to use Tableau Prep, Python and Alteryx to create the tables. See the list below to understand which of the tables are to be created by Python/Alteryx and which ones are to be created by Tableau Prep.

Using Python and /or Alteryx, create jobs to populate the fact and dimension tables identified above by extracting the data from the Dillard’s relational database that you implemented in your LMU Build and using the **stores.csv and sku.csv** tables even though they exist in your LMU Build database. In real life, the data sources can be in different areas and you have to integrate them into a single repository:- the function of ETL.

*Alteryx/Tableau Prep*

Create the dim\_sku table in your destination server (AWS or build) using Alteryx. Note that department names should be part of this table. Thus, you have to join two tables in your Alteryx job.

Using appropriate logic, create all the other dimension tables using Alteryx or Tableau Prep.

*Python*

Create the return fact table in your destination server. You need to use aggregation in the DataFrame (or use the sql to create the aggregate and then execute it in pandas or dask) along with other code that you have learned in this course.

Create the final fact table (for non-returned sales) in your destination server. Use aggregation in the dataframe (or use the sql to create the aggregate and then execute it in pandas or dask) along with other code that you have learned in this course. Note that you have to get the cost information for each of the sku (and the cost may be based on a sku-store combination) to calculate the profit for each of the transactions and then aggregate that at the correct grain level.

6. **Using the Data Warehouse**

Provide the following for each data request specified in #3 above (including the ones that you came up with):

1. SQL query to SELECT from your data warehouse
   1. Save all SQL statements in a single file named ***assignment\_02.sql***. Add a comment above the SQL to denote the task, i.e. # Strategic 2, # Analytical 3.
2. Visualize the results with Tableau and/or Power BI. Create a dashboard for each data request category (Strategic, Operational, Analytical). Save the dashboard and its visualizations into its own Tableau Packaged Workbook (for Tableau). (File -> Export Packaged Workbook). The PowerBI will save in its own format (if used).

7**. Implementing the Star Schema in PowerPivot**

Using the final Datawarehouse (in whatever server you have decided to have it) and ODBC connections, create the star schema in PowerPivot. Develop appropriate pivot tables to get some of the information mentioned in part 3.

**Deliverables (all files should be preceded with your initials):**

***star\_schema\_details.xlsx***

***data\_warehouse\_erd.jpg***

***assignment\_2.sql***

***file with the visualizations***

***All the Alteryx and Tableau Prep files***

***Python code in Jupyter Notebook format***



**Dillard’s Relational Database Entity Relationship Diagram**

