BI Course Project

Business Intelligence Spring 2015
Due Date – No later than midnight Sunday May 3, 2015
This project must be completed in project teams as assigned by the instructor.

INTRODUCTION

To goal of this project is to design and implement a data warehouse for the BIke Project Company (hereafter BIke). BIke executives have declared that a new data warehouse is to be the center piece of their *Data Driven Decision Making* strategic initiative and must be capable of answering typical business intelligence (BI) questions

This project is broken into two parts. In part 1, you will use Microsoft's SQL Server Integration Services (SSIS) to construct a repeatable ETL process to move data from several data sources into a dimensional model instantiated as a SQL Server relational database. In part 2 of this project, and in keeping with current best practices, you will access the completed data warehouse tables via an OLAP layer implemented using Microsoft's SQL Server Analysis Services (SSAS).

PART 1 – DATA WAREHOUSE AND ETL REQUIREMENTS (50%)

An objective of any ETL process is that the ETL **processes be repeatable**. To that end the IT department at BIkes, has decided to implement the initial ETL load processes using Microsoft's SQL Server Integration Services (SSIS).

GENERAL REQUIREMENTS

To implement the ETL processes you will use the BIke's OLTP database schema as your operational DB schema guide. Please refer to the BIkes schema document (*Schema_BIkeProject_OLTP.PDF*) located in the Blackboard Course Project folder.

Note that not all process and data requirements are explicitly laid out below. It is very common for some processes to be inferred by the data requirements. For example, the Date dimension requirements will require ETL processes to bring the requisite data into existence even though there is no corresponding date-related OLTP table.

DATA WAREHOUSE DATA REQUIREMENTS

Your DW must be able to support queries regarding **sales**, **time, products, sales people, stores, and customers**. Your design should follow the best practices as recommended by Kimball and Ross. I.e., the dimensions derived from operation and external data sources should be fully "de-normalized" and in first normal form. Snowflake designs are discouraged.

Specific Requirements

1. Model data warehouse *logical* data design using an ERD
    - Your DW ERD must be complete and professionally rendered dimensional model. I.e., using Visio or other drawing tool.
    - Normally, as you know, the list of attributes to be included in you dimensional model would follow from a thorough requirements gathering in process. For the purposes of this project you may assume that that process has been carried out and the results are reflected in the Appendix to this document. Said differently, I'm giving you the list of attributes to be included in each of the tables in your data warehouse. Your task is reduced to sourcing the attributes as required.

- Your ERD schema must clearly show all table attributes with any primary keys, foreign keys and degenerate dimensions clearly labeled. For a design example, refer to the Warehouse schema shown in chapter 3 of the Kimball text.

2. Construct Routine Dimensions

The following table describes the required dimensions and their likely sources.

Dimension Table Name	Source Tables	Special Instructions:
DimProduct	oltp.Product oltp.ProductSubcategory oltp.ProductCategory	
DimCustomer	oltp.Customer oltp.EmailAddress oltp.PhoneNumber oltp.Address oltp.State	Keep only USA customers. CountryCode='US' Retain only "Home" *AddressType* in DimCustomer. Retain 'Home' phone types in DimCustomer. PhoneType='Home' Retain 'Cell' phone types in DimCustomer. PhoneType='Cell'
DimStore	oltp.Store oltp.Region	Add ID 9999 for Internet sales. See below.
DimSalesPerson	oltp.SalesPerson	Add ID 9999 for Internet sales. See below.

Table1 Notes: the oltp prefix on table names indicates the source of the table is the BIkes OLTP database.

Note that all dimensions should be fully "de-normalized" and in first normal form. I.e., no snowflaking. You must include all attributes for each dimension table as shown in the Appendix. In cases where the attributes for a dimension are assembled from more that 1 source, you will be required to "flatten" the source tables to produce a dimension table that is in 1NF.

Several years ago, BIke's opened several stores outside of the U.S.A. These international stores have since closed, although the sales data from these stores still permeates the OLTP system. Since these international stores no longer exist, your *Customer* dimension should only retain customers from the United States.

About the time the international stores close, BIke's opened an online store. Since money was tight, the online sales were made to work within the BIke's existing OLTP framework. Each online sale is recorded in the SalesOrder table with the *OnlineOrderFlag* set to "on". Likewise, these sales have *StoreID* and *SalesPersonID* set to NULL. Management is interested in comparing Internet sales against their brick-and-mortar stores. This will require that *StoreID* and *SalespersonID* be set in the fact table to point to a fictitious store and salesperson in the DimStore and DimSalesperson tables respectively. The Internet store should have a *StoreDescription* set to "Internet" while Internet *SalesPerson* should have a last name set to "Internet". To make things easier, both IDs should be set to a well known value such as *9999*.

3. Construct a Date Dimension

Your Date dimension must be named **DimDate** and should be modeled at the level of a single day. Time should begin with the first day of the earliest order date and should end 5 years hence. Your Date dimension must support the attributes defined in the Appendix.
Regarding the select US Federal holiday attributes – xmasFlag, IndepDayFlag and NewYrsFlag. If a holiday falls on a Saturday it is celebrated the preceding Friday; if a holiday falls on a Sunday it is celebrated the following Monday

While it is entirely possible to craft a date dimension using SSIS, the best idea is to use Excel. Your strategy would be to create all of the columns (attributes) and rows (days) required in Excel. Once compete, save a copy of the worksheet as a CSV file type for easier import into SSIS. You can import Excel files into SSIS; however, you may find that it is difficult to change the data types to match those in your *DimDate* data definition language statements (DDL).

4. Construct a Fact Table

The "grain" of facts you capture must support the design objectives of the data warehouse. Best practice suggests capturing facts at the level of an OLTP transaction.

Fact Table Name	Source Tables
FactStoreSales	oltp.SalesOrder oltp.SaleOrderDetail

You must include all of the attributes shown in Appendix in your fact table. As mentioned above, several years ago, Bike's opened several stores outside of the U.S.A. Since these international stores no longer exist, your fact table should only retain sales for customers from the United States.

Lastly and as previously mentioned, the *SalesOrder* table contains Internet sales. All facts having their *OnlineOrderFlag* set to "on" should have both *StoreID* and SalesPersonID attributes "transformed" (set) to 9999.

ETL PROCESS REQUIREMENTS

BIkes outsources their marketing function. The marketing firm used by BIkes uses Survey Monkey and legacy CRM application to record customer responses to post-sales surveys. Unfortunately, not being DB experts, the only way they can make this information available to BIkes is in a comma-separated-values text file named *SalesReasons.csv*. Since this information is updated regularly, you have agreed to access this data via the marketing firm's FTP server as part of your ETL control flow processes. Of course, not every customer will have responded to the survey; however, Bike's management would like to have these sales reasons associated with each customer where such data exists. Your ETL process must import this file and associate the sales reasons with the correct customer. The file contains 1 line per customer for each customer that has responded to a post-sales survey. Every customer listed in the file will have primary sales reason for buying from BIkes (the *PrimarySalesReason* column). Some customers, but not all, will have both a primary and a secondary sales reason (the *SecondarySalesReason* column). You must record the primary and secondary reasons as part of **DimCustomer**. These attributes should be named *PrimarySalesReason* and *SecondarySalesReason* respectively.

Accessing the SalesReasons File

You must access the sales reason file via the marketing firms FTP server as part of the control flow used to build your customer dimension table. You can use the following information to create the necessary connection to the FTP server via the SSIS *FTP Connection Manager* used in conjunction with the *FTP Task*. The SSIS *FTP Task* makes use of

an *FTP Connection Manger* to access and download information from an FTP server. Name your FTP Connection *SalesReasons.* The FTP Connection Manager Parameters are as follows.

Server name	ftp.drivehq.com
Server port	21
User name	robertsj1503
Password:	duqbisp15
Use passive mode	Yes

As for the SSIS FTP Task, I expect for you to be able to read and make use of the relevant documentation. **Note**: as part of normal processing, the FTP Task will download the sales reason file each time the task is executed. You must request that the file be downloaded (stored) into the *Datasources* folder as described below. Once successfully downloaded, the file will be available to any subsequent control or data flow elements.

For planning purposes you may view the file from a browser using the following URL
ftp://robertsj1503:duqbisp15@ftp.drivehq.com/DQUBISP15/SalesReasons.csv

Data Quality of the SalesReasons File

A quick check of the sales reasons file will lead you the correct conclusion that this data has quality issues. Specifically, there are 2 types of quality problems (1) duplicate data and (2) typographical errors. To correct these errors you will use the **Fuzzy Grouping** transform and the **Data Cleansing** transforms of the SSIS. As in our lab work, the **Data Cleansing** transform assumes you have access to a pre-existing knowledge base suitable for cleansing your data. Similar to our lab projects, you will need to build this simple knowledge base using the **SQL Server Data Quality Client**.

Using an approach that very closely resembles the activities in the data quality labs, you will:

1. Construct a knowledge base capable of correcting any and all existing typographical errors. Note that typographical errors are restricted to the primary and secondary sales reason columns.
2. Export your published the knowledge base to your project's **DataSources** folder as you will (most likely) need to re-create this knowledge base when you move your project between machines.
3. Use your new knowledge base as the basis for using the **Data Cleansing** transform.
4. Use **Fuzzy Grouping** transform to identify and eliminate duplicate records from the sales reasons file. To keep this simple, you may assume that records that are duplicates have the same values for all attributes in the SalesReasons.csv file. I.e., the CustomerID, PrimarySalesReason, and SecondarySalesReason attributes. You should consider any record as a duplicate of another record when the record's (a) **Fuzzy Grouping** transform *_key_in* attribute is not equal to the *_key_out* attribute and (b) **Fuzzy Grouping** transform *_score* attribute value exceeds .80 (80%).

PART 1 DELIVERABLES

1. **Logical** DW ERD (i.e., star diagram) in PDF format
2. Functioning SSIS ETL project in ZIP file format. Your folder structure should be as follows:
   YourGroupName
   SSIS_ETL_Project (subfolder)
   DataSources (subfolder)
   Scripts (subfolder)

SSIS_ETL_Project subfolder should contain the SSIS project files. The *DataSources* subfolder should contain all data required by your project. I.e., the sources OLTP database files (.MDF, .LDF), data warehouse database files (.MDF, .LDF), the Marketing department CSV file, and an exported knowledge base. The *Scripts* folder should contain all of the scripts containing the SQL DDL statements used to create your data warehouse dimension and fact tables.