Data Platform Implementation

Assignment 2 focuses on the development of data platforms. Therefore, this assignment is a “hands-on” exercise where you will create a “classic” data mart for the organization, develop ETL processes to load data from the OLTP environment, set up a ROLAP cube, and run on it a set of queries that revolve around important business questions.

Table of Contents

Preface: Instructions & Guideline 2
Data Warehousing Implementation [30 points] 3
1. Landing/Staging Area [3 points] 5
 1.1 Landing area table creation scripts 5
 1.2 Create a Data Integration Repository 6
 1.3 Create PDI job to create tables of the BI_Bikes_<groupNo> database 7
 1.4 Create PDI job to populate tables of the BI_Bikes_<groupNo> database 8
 1.5. PDI jobs parametrization 8
 Deliverables 9
 Resources 9
2. Data Mart [15 points] 10
 2.1 Data Mart creation 12
 2.2 ETL implementation 12
 2.3 ROLAP Cube definition 13
 Deliverables 14
 Resources 14
3. Business Analytics (Queries) [12 points] 15
 3.1 Sales – profit for top product categories 15
 3.2 Sales – revenue for countries 15
 3.3 Sales – most profitable customers in the time period 15
 3.4 Sales – most actively purchasing customers 16
 3.5 Logistics - monthly amounts of shipping costs for the customers of a country 16
 3.6 Production – most sold product models in each of the categories 16
 3.7 Accounting – most profitable countries for the goods in the price range 16
 3.8 Accounting – taxes charged from the customers in certain countries 17
 Deliverables 17
 Resources 17
Bonus question [up to 3 points] 17
Appendix: Data Dictionary 18
Appendix: Folder structure for the assignment deliverable. 22

Preface: Instructions & Guideline

Deadline: Make sure to upload all your results **before the end of December 5, 2023.**

Submission guidelines: We are going to follow a number of naming conventions throughout the assignment. Please **pay attention to them and follow carefully**, as this will ensure your code executes correctly in the test environment.

Compress your solution directory and upload it on TUWEL using the following filename:

```
BI_2023_Assignment_2_Group_<groupNo>.zip/.tgz
```

Within this ZIP-file, the top-level directory is named BI_Projects. Inside this directory there is a folder named BI_<groupNo>, **please replace** <groupNo> **with your actual Group Number**, with the leading zero.

This BI_<groupNo> folder contains all your deliverables as follows:

1. `csv_data` folder that contains csv files with source data
2. Folders named `task1`, `task2`, `task3` for individual tasks.
3. A text file group`_<groupNo>.txt` that lists your group members (name, student id)
4. Possibly a text file `comments_<groupNo>.txt` with any comments that you might want to make. Also, place the answer to the bonus question(s) into this file.

Each group member should upload a submission. In the best-case scenario, when members of a group work on the assignment together as a team, then for the two members of the same group the contents of their submission will be identical.

The final directory layout of the extracted solution should look as follows:

```
BI_Projects
  \-BI_XX
    |
    +---csv_data
    |       Address.csv, Product.csv, SalesOrderHeader.csv, ...
    |
    +---task1
    |   \---oltp_db
    |           TB_Address.sql, TB_Product.sql, TB_SalesOrderDetail.sql ...
    |
    +---task2
    |   +---data_mart
    |   |       Dim_Customer.sql, Dim_Product.sql, Fact_InternetSales.sql ...
    |   |
    |   +---etl_sales
    |   |       *.ktr / *.sql
    |   |
    |   \---olap_cube
    |           bike_sales.xml
    |
    +---task3
    |       5_1.mdx / 5_2.mdx / 5_3.mdx ...
    |       5_1.sql / 5_2.sql / 5_3.sql ...
    |   1 - Create Bikes.kjb
    |   2 - Load Bikes.kjb
    |   3 - Create BikesDW.kjb
    |   4 - Load BikesDW.kjb
    |   BI_Bikes_XX.kdb
    |   BI_BikesDW_XX.kdb
    |   comments_XX.txt
    |   group_XX.txt
    |   repository.log

* Please make sure to replace XX with Your Group Number
```

Questions. Please post general questions in the TUWEL discussion forum. You can also discuss problems and issues you are facing there. We appreciate it if you help other students tackle general issues or questions (and may take that into account if you are short a few points for a better grade). For obvious reasons, however, please do not post any solutions there.

For specific questions regarding the assignments, you can contact our tutors

- Ildar Fatkullin (ildar.fatkullin@tuwien.ac.at)
- Peter Lahnsteiner (peter.lahnsteiner@tuwien.ac.at)

Or in case of other issues

- Katja Hose (katja.hose@tuwien.ac.at).

Data Warehousing Implementation [30 points]

Preliminaries. You will use a set of open-source, cross-platform tools to implement a data mart and use it to address a range of business questions:

- Java Development Kit (JDK)
- Java Runtime Environment (JRE)
- Pentaho Data Integration (PDI)
- Pentaho Schema Workbench (PSW)
- MySQL
- MySQL Connector/J

The complete software stack is available for Linux, Windows, and macOS. Compared to more full-fledged commercial BI solutions, the individual elements can be set up quickly on any machine with minimal system requirements. Our exercise focuses on developing an understanding of concepts rather than learning a particular vendor's graphical tools – the stack serves this purpose well. Please refer to the "Lab Setup" section of this document for installation instructions.

Introduction. In this assignment, we use a fictional bicycle wholesaler company named "Big Time Bikes". The Company has for sale bike components, accessories, clothing, and many different brands of bikes grouped into three categories - mountain bikes, road bikes, and touring bikes.

"Big Time Bikes" serves customers globally, including Australia, Canada, France, Germany, the United Kingdom, and the United States. One of the business models used in "Big Time Bikes" is internet sales that serves individual customers.

"Big Time Bikes" uses a transactional database and data warehouse to support their business. These database systems cover sales, material management, production, finance, and human capital management business processes. However, we consider only a part of the system related to internet sales as a case study to develop a self-service BI system.

In a pilot project, you are tasked with developing a data mart that provides detailed access to sales data, allows the management team to analyse critical business questions, and supports their decision-making. In this pilot project, you will:

1. Create a landing/staging area and populate this database with the data from OLTP database.
2. Create an "Internet Sales" data mart that uses star schema.
3. Define and execute an ETL process to extract, transform, and load data from the landing/staging area into the Data Mart.
4. Create an OLAP cube for multidimensional sales data analysis.
5. Implement a set of queries to inform management about critical business metrics.

Project description. A data mart is a subject-oriented database that is often a partitioned segment of an enterprise data warehouse. The subset of data held in a data mart typically aligns with a particular business unit like sales, finance, or marketing. Data marts accelerate business processes by allowing access to relevant information in a data warehouse. Because a data mart only contains the data applicable to a certain business area, it is a cost-effective way to gain actionable insights quickly.

In our case our data mart stores data on "Internet sales" business area.

Schematically the architecture that your team is tasked to implement may be depicted as follows:

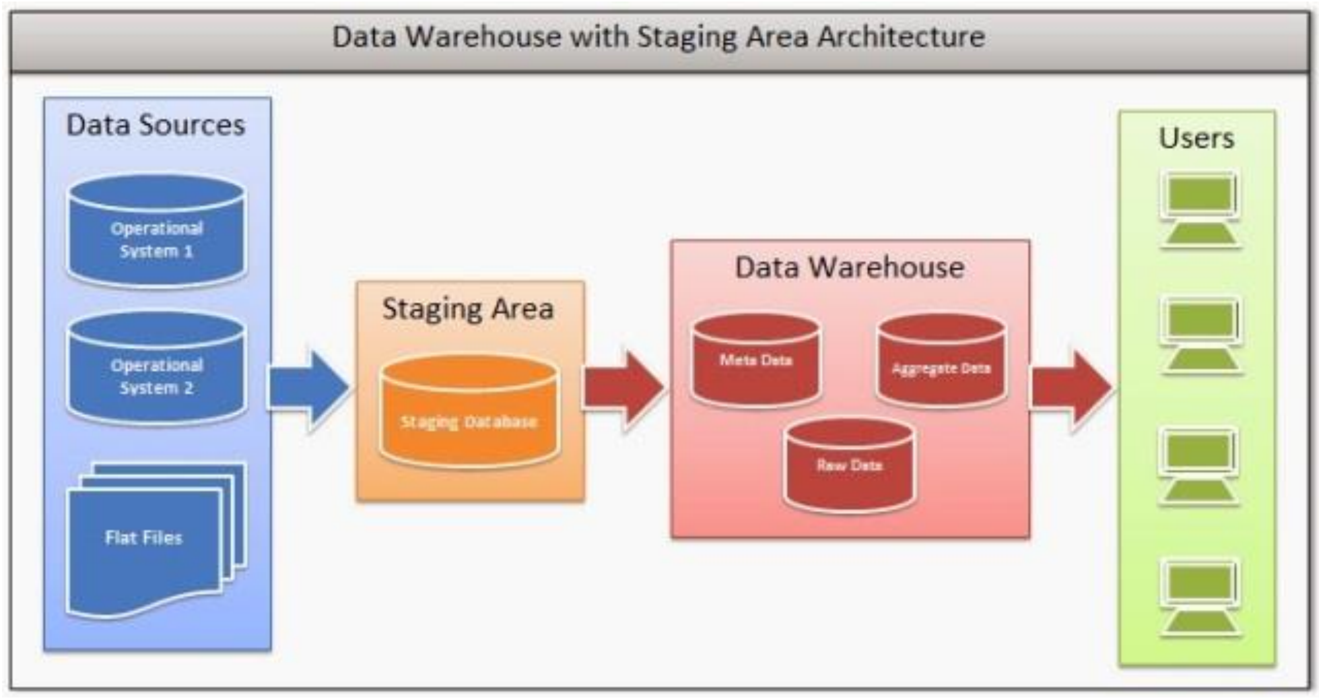


Figure 2. Data Warehouse Arcchitecture

In our case, the Staging Database is represented by the BI_Bikes_XX database. Data Warehouse is represented by BI_BikesDW_XX database.

1. Landing/Staging Area [3 points]

The data delivered to you from the Company's OLTP database has the following schema, Figure 3.

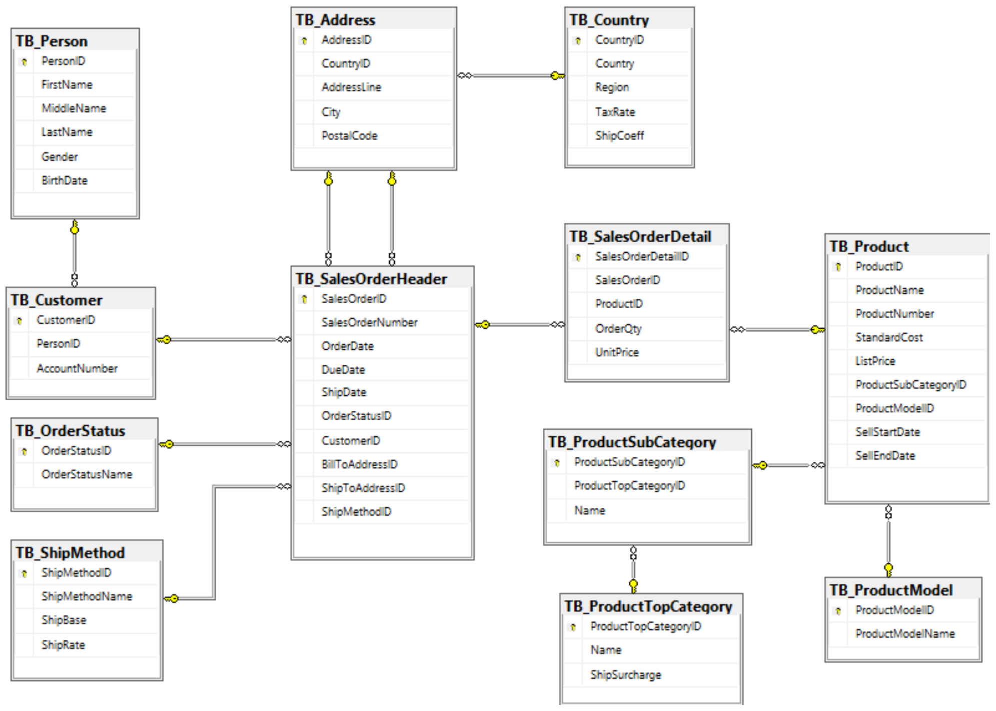


Figure 3. Logical schema of a portion of OLTP

Based on the logical relational data model depicted in Figure 3, implement a Landing/Staging area and populate it with the data coming out of the OLTP system (see also Appendix: Data Dictionary).

We already created a database named BI_Bikes_<groupNo> (see 1.6.2.). This database will play a role of a Landing/Staging area. The source data is delivered to you in the form of CSV files.

1.1 Landing area table creation scripts

For each table, create a sql script **TB_<table_name>.sql** that creates a table according to the definition specified in the Appendix: Data Dictionary.

Define necessary primary keys as table constraints and name them `PK_<table_name>`, where `<table_name>` is replaced with the respective name of the table, e.g., for table `TB_Address` the primary key is named `PK_Address`.

Define necessary foreign keys as table constraints and name them `FK_<column_name>_<tableName>`, where `<column_name>` is the name of the column without the 'ID' suffix that represents the reference and `<tableName>` is the name of the table that stores the foreign key – for example, the references between `TB_SalesOrderHeader` and `TB_Address` are represented by the columns `BillToAddressID` and `ShipToAddressID`. The corresponding foreign key constraints should be named `FK_BillToAddress_SalesOrderHeader` and `FK_ShipToAddress_SalesOrderHeader`, respectively (this should ensure that the foreign key names are unique).

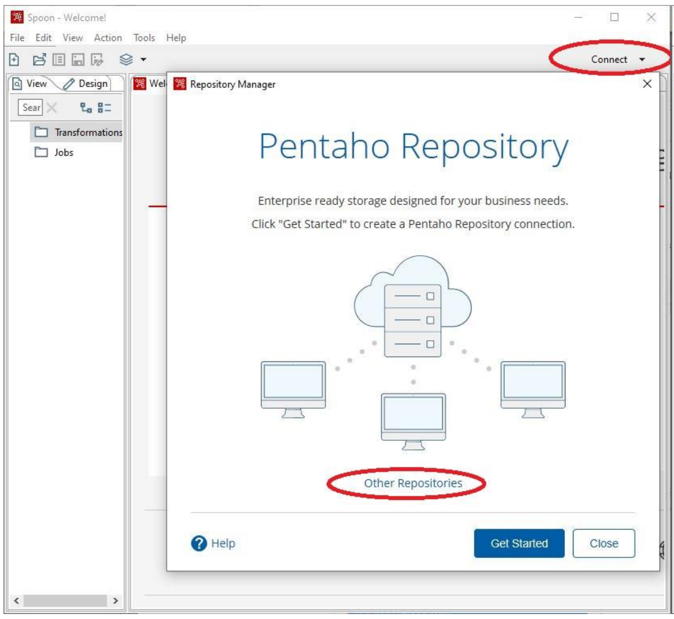
Place the scripts in your solution directory:

`BI_Projects/BI_<groupNo>/task1/oltp_db/TB_<tableName>.sql.`

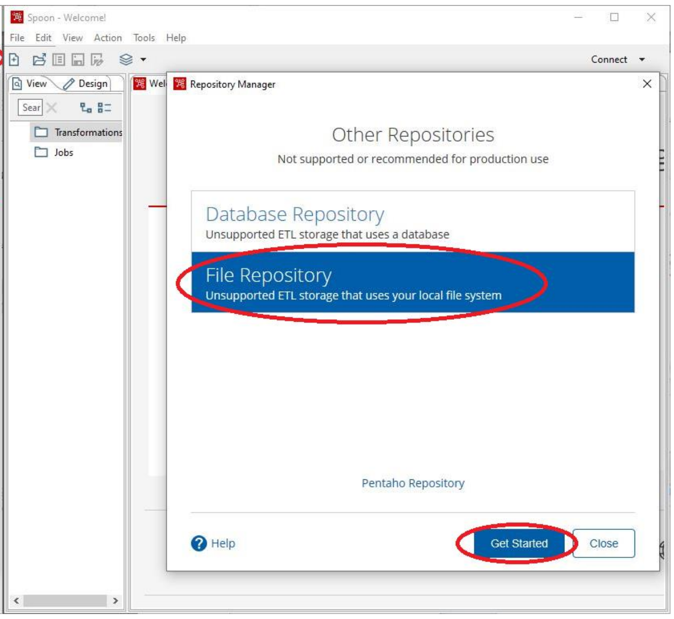
1.2 Create a Data Integration Repository

Launch Pentaho Data Integration (PDI) by executing Spoon.bat/spoon.sh. Spoon is the graphical interface tool to create transformations and jobs.

- Press Connect and click "Other Repositories" button to add a repository
- Choose File repository and select **`BI_Projects/BI_<groupNo>`** directory and assign BI_<groupNo> as a name (where <groupNo> is your group number)



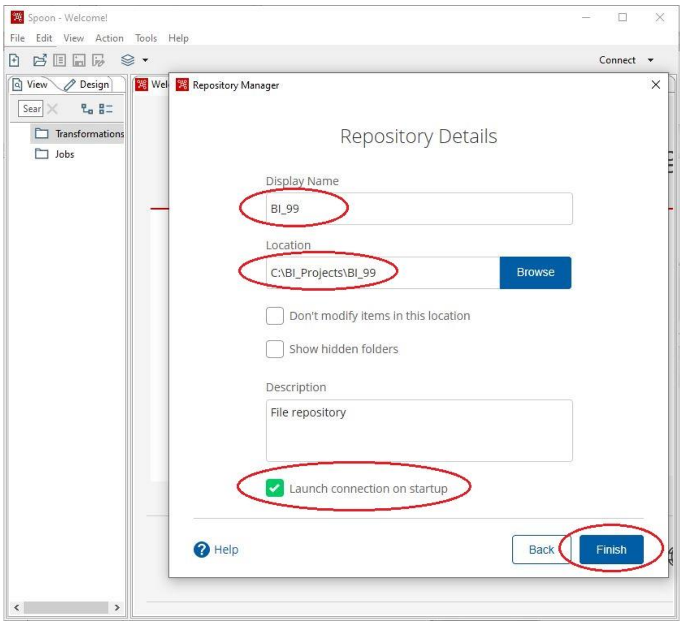
Enterprise ready storage designed for your business needs.
Click "Get Started" to create a Pentaho Repository connection.



Not supported or recommended for production use

Database Repository
Unsupported ETL storage that uses a database

File Repository
Unsupported ETL storage that uses your local file system



Display Name: BI_99
Location: C:\BI_Projects\BI_99
☐ Don't modify items in this location
☐ Show hidden folders
Description: File repository
☑ Launch connection on startup

1.3 Create PDI job to create tables of the BI_Bikes_<groupNo> database

Create a job in PDI client (Spoon) to invoke all table creation scripts from 1.1. in the appropriate sequence.

a. Create an integration job and name it "1 - Create Bikes". Save it to your **BI_Projects/BI_<groupNo>** directory.

b. Create Database connection to your BI_Bikes_<groupNo> database. This database connection will be re-used in other jobs and transformations.
In the General tab:
   - Connection name: BI_Bikes_<groupNo>
   - Connection type: MySQL
   - Access: Native (JDBC)
   - Host Name: localhost
   - Database Name: BI_Bikes_<groupNo>
   - Port Number: 3306
   - Username: **bi2023**
   - Password: **bi2023W!**

   In the Advanced tab:
   - Paste the following code snippet to execute right after connecting:
   - `SET GLOBAL local_infile = 'ON';`
     (This will enable the server-side of "loading data from local files" capability).

   In the Options tab:
   - Create a new parameter, name it `allowLoadLocalInfile` and set it to "true".
     (This will enable the client-side of "loading data from local files" capability).

   You can also now check your connection by clicking "Test" button here.

c. Add [Start] step.

d. Add the SQL create scripts to the integration job using [Scripting/SQL] steps in an appropriate sequence.

e. Edit the [Scripting/SQL] step(s)
   - Double click to edit the step
   - In the connection section, choose the database connection we created earlier (see 1.3.b).
   - Check "SQL from file".
   - Choose the path to SQL file(s) you created earlier (see 1.1)

f. Connect all job entries with hops, starting with [Start] job entry.

g. Execute the integration job to create all tables of the BI_Bikes_<groupNo> database.

1.4 Create PDI job to populate tables of the BI_Bikes_<groupNo> database

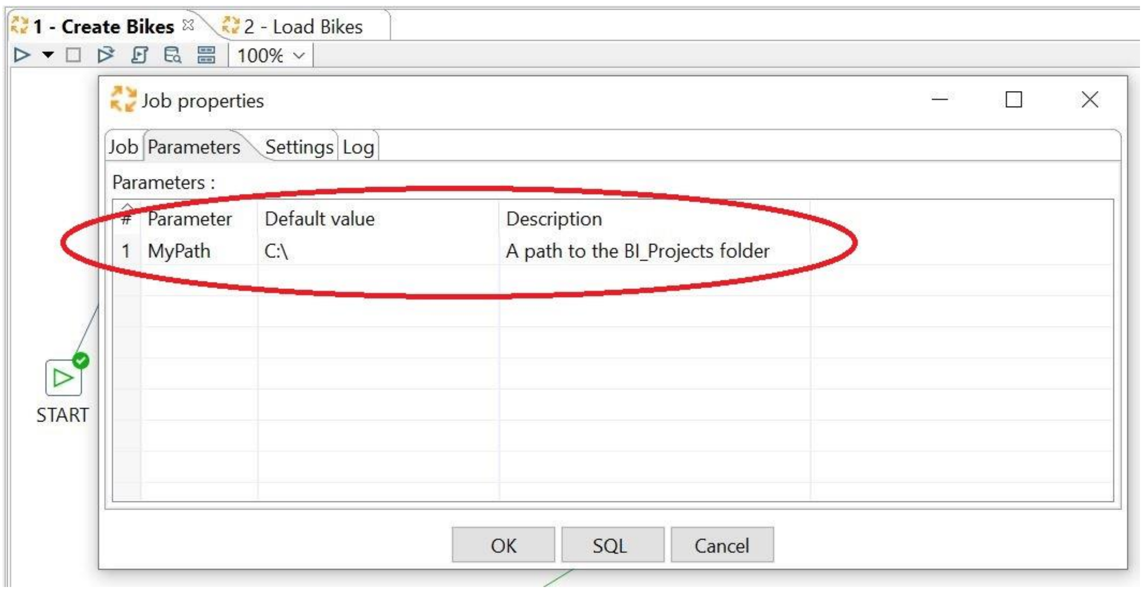
The data that comes from the OLTP database has been provided in the form of CSV files (tab-separated files). To load the data into your staging database, create a new integration job named "2 – Load Bikes", add the database connection, and use "Bulk load into MySQL" steps to load the data into MySQL tables in the appropriate sequence (i.e., ensure that the database is in a consistent state after loading).

Note: The provided CSV files are tab-delimited. Therefore, when setting up the "Bulk load into MySQL" steps, remove the default comma from the delimiter field and click "Insert TAB". Make sure to load the data into the correct columns.

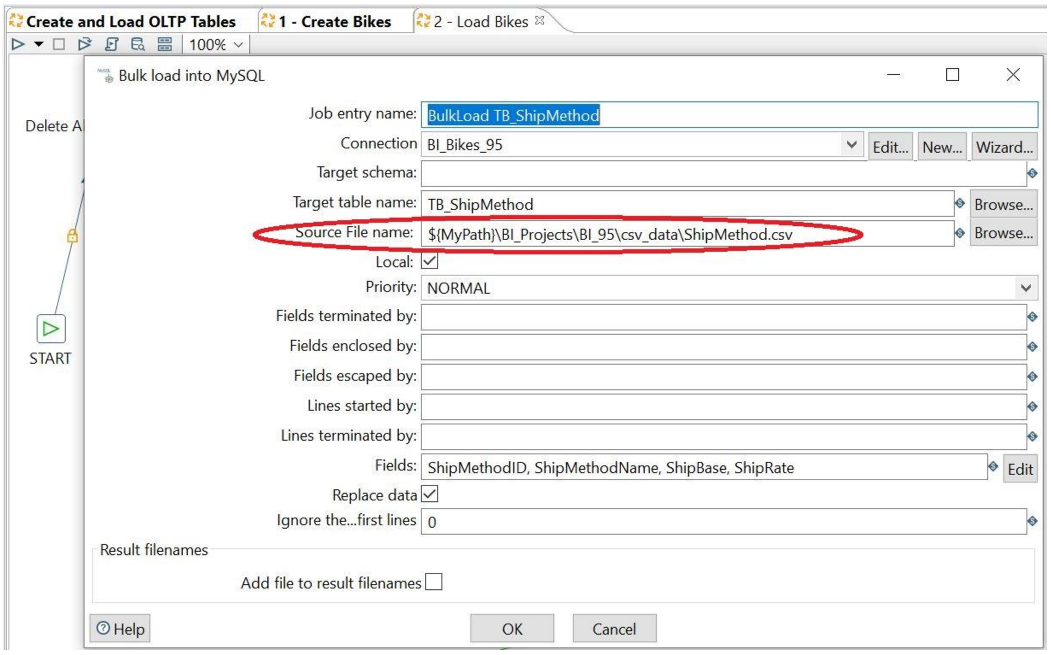
1.5. PDI jobs parametrization

The use of PDI job parameters is recommended to make it easier to work on Data Integration jobs as a group. This is a short manual on implementing a job parameter for a file path to CSV files.

1. In your existing, or newly created job, double-click anywhere on the canvas, or right-click on the canvas and choose "Properties".
2. Go to "Parameters" tab and create a new parameter, name it "MyPath" or any other name you like.
3. Give this parameter a default value, for example, "C:\" (without the quotes) - this is the location of your "BI_Projects" folder (see screenshot).
4. Now you can use this parameter in this job's entries as a part of your file path (see screenshot for an example).
5. Each time you run this job you can provide for this parameter a value that is different from the default value (see screenshot).

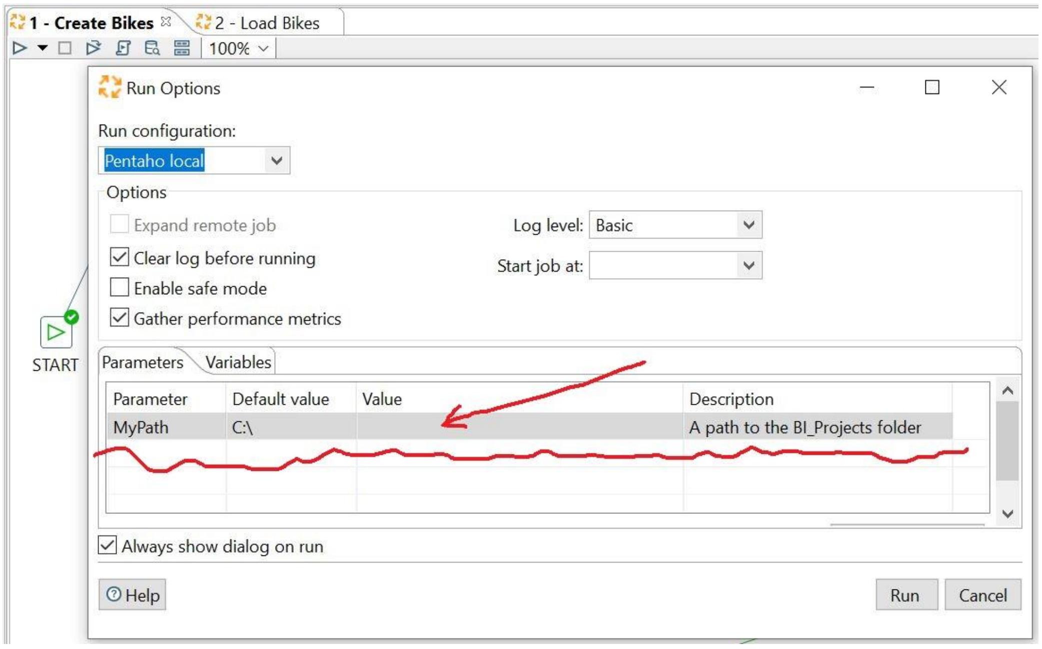


#	Parameter	Default value	Description
1	MyPath	C:\	A path to the BI_Projects folder



Bulk load into MySQL

Job entry name:	BulkLoad TB_ShipMethod
Connection	BI_Bikes_95
Target schema:	
Target table name:	TB_ShipMethod
Source File name:	${MyPath}\BI_Projects\BI_95\csv_data\ShipMethod.csv
Local:	✓
Priority:	NORMAL
Fields terminated by:	
Fields enclosed by:	
Fields escaped by:	
Lines started by:	
Lines terminated by:	
Fields:	ShipMethodID, ShipMethodName, ShipBase, ShipRate
Replace data	✓
Ignore the...first lines	0
Add file to result filenames	☐



Run Options

Run configuration: Pentaho local

Options
- ☐ Expand remote job
- ✓ Clear log before running
- ☐ Enable safe mode
- ✓ Gather performance metrics

Log level: Basic

Start job at:

Parameter	Default value	Value	Description
MyPath	C:\		A path to the BI_Projects folder

✓ Always show dialog on run

Deliverables

In this first task your **BI_Projects/BI_<groupNo>/task1/oltp_db** folder now contains table creation scripts, the **BI_Projects/BI_<groupNo>/** folder contains two Kettle job files (.kjb extension) and one file that stores database connection info (.kdb extension).

Resources

Pentaho Data Integration Tutorials:

- https://help.hitachivantara.com/Documentation/Pentaho/9.0/Products/Job_entry_reference
- https://help.hitachivantara.com/Documentation/Pentaho/9.0/Products/Transformation_step_reference

2. Data Mart [15 points]

Your team has been tasked with implementing a sales analysis data mart as the first step towards the development of a comprehensive business intelligence system for the Company. To this end, you have conducted extensive requirements elicitation interviews with the responsible managers of the sales, logistics and products departments. Your task now in this step is to implement the corresponding star schema in a separate database (BI_BikesDW_XX) and the process that extracts, transforms, and loads the data from the staging area database (BI_Bikes_XX).

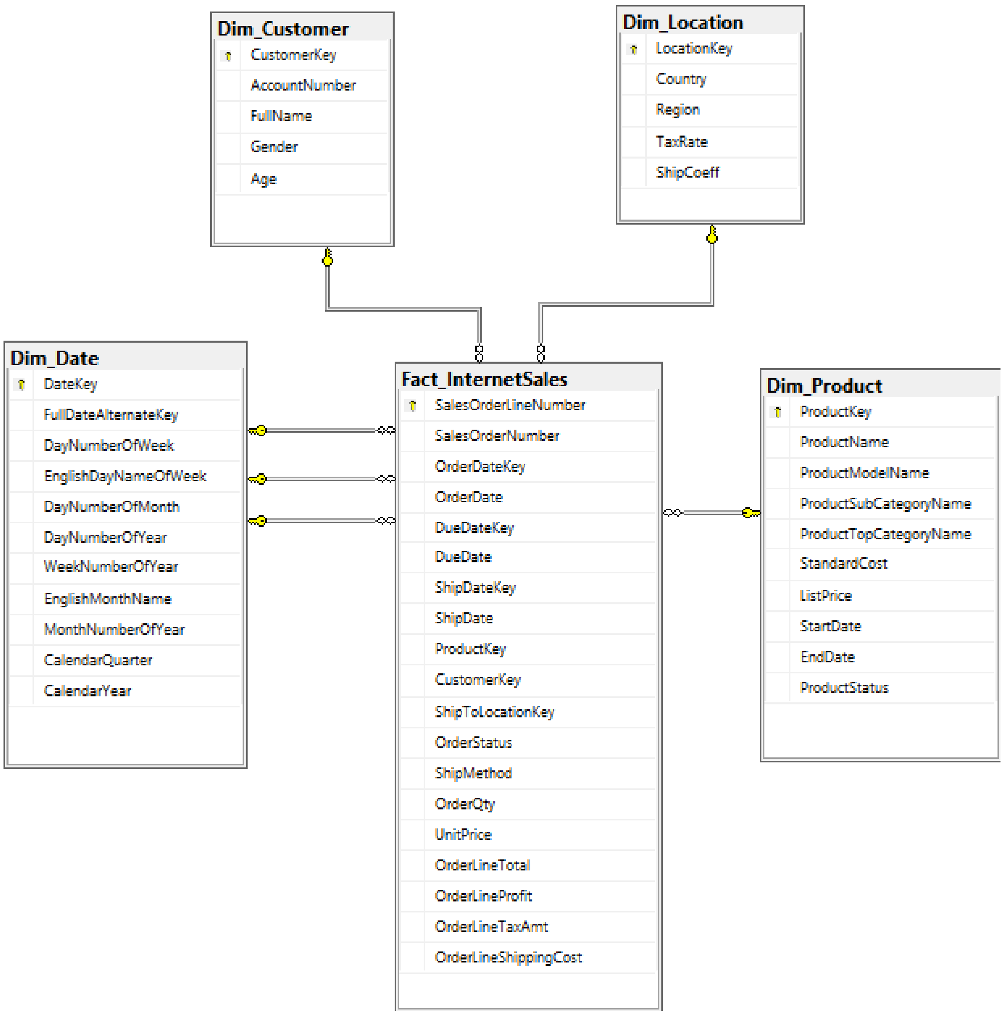


Figure 4. Logical schema of a required data mart database

In the following, you will find the specification of the required data mart database that should be implemented in a star schema (Figure 4), as well as a specification of the transformations that need to be performed (Table 1).

Most of the field names in the Star schema are self-explanatory, but some fields require comments. Please refer to Table 1.

Attribute	Datatype	Transformation Instruction
CustomerKey	INT	= CustomerID
LocationKey	INT	= CountryID
ProductKey	INT	= ProductID
DateKey	INT	YEAR(@date) * 10000 + MONTH(@date) * 100 + DAYOFMONTH(@date) AS DateKey
FullName	VARCHAR(150)	Concatenate first, middle and last name to form a customer's full name in the following format: *FullName = FirstName MiddleName LastName*
Age	INT	Using a customer's birth date, calculate the age as of 30th of September 2021 (use '2021-09-30' as the end date for the interval calculation).
SalesOrderLineNumber	VARCHAR(50)	The Sales Order Line Number is built by concatenating *'SOL' + SalesOrderID + '-' + SalesOrderDetailID* Note that the resulting string must not contain any whitespaces. E.g., "SOL45021-5135"
ProductStatus	VARCHAR(50)	If SellEndDate is NULL or SellEndDate is later than '2021-09-30' then 'Current', otherwise SellEndDate is 'Discontinued'.
OrderDateKey	INT	10000 * Year(OrderDate)+ 100 * Month(OrderDate)+ DayOfMonth(OrderDate)
DueDateKey	INT	10000 * Year(DueDate)+ 100 * Month(DueDate)+ DayOfMonth(DueDate)
ShipDateKey	INT	10000 * Year(ShipDate)+ 100 * Month(ShipDate)+ DayOfMonth(ShipDate)
OrderStatus	VARCHAR(50)	Use the order status name.
ShipMethod	VARCHAR(50)	Use the shipment method's name.
OrderLineTotal	DECIMAL(13,4)	calculate the total revenue per order line with the formula *OrderLineTotal = OrderQty * UnitPrice*
OrderLineProfit	DECIMAL(13,4)	calculate the profit per order line with the formula *OrderLineProfit = OrderLineTotal – OrderQty * StandardCost*
OrderLineTaxAmt	DECIMAL(13,4)	calculate the tax amount of an order line with the formula *OrderLineTaxAmount = OrderLineTotal * TaxRate*
OrderLineShippingCost	DECIMAL(13,4)	calculate the shipping costs for each line as follows: *OrderLineShippingCost = ShipSurcharge + ShipBase + (OrderQty * ShipRate * ShipCoeff)*

Table 1. OLTP to OLAP Transformations

2.1 Data Mart creation

We already created a database named BI_BikesDW_ (see “Lab Setup”). This database will play a role of a Data Mart.

Similar to “1.1. Landing area table creation scripts”, for each table of the Data Mart, create a sql script **`Dim_<table_name>.sql/Fact_<table_name>.sql`** that creates fact and dimension tables according to the definition specified in the Appendix: Data Dictionary.

- Name the fact table `Fact_InternetSales`
- Name the dimension tables `Dim_<dimension_name>`

Define necessary primary keys as table constraints and name them `PK_Dim<table_name>` or `PK_Fact<table_name>`, where `<table_name>` is replaced with the respective name of the table without the prefix, e.g., for table `Dim_Customer` the primary key is named `PK_DimCustomer`.

Define necessary foreign keys as table constraints and name them `FK_<column_name>_<table_name>`, where `<column_name>` is the name of the column without the 'Key' suffix that represents the reference and `<table_name>` is the name of the table that stores the foreign key – for example, the reference between `Dim_Customer` and `Fact_InternetSales` is represented by the column `CustomerKey`. The corresponding foreign key constraint should be named `FK_Customer_FactInternetSales`.

Place table creation scripts to the following folder: **`BI_Projects/BI_<groupNo>/task2/data_mart/`** Create a data integration job that executes the create scripts in an appropriate sequence (see 1.3). Name your job "3 - Create BikesDW" and run it to create the data mart.

2.2 ETL implementation

Once the data mart database has been created, your team will develop the required ETL processes that extract the data from the staging area database, transform the data according to transformation instructions specified in Table 1, and load the data into the data mart's fact and dimension tables.

As in the previous steps, you will use Pentaho Data Integration's Spoon to define the ETL process. For the actual transformations, you have two options: you can either specify the transformations in plain SQL or define the ETL jobs directly in Spoon, using the built-in transformation steps. In either case, you should create a data integration job that loads the data into the data mart in an appropriate sequence to ensure consistency.

Although it is possible in our simple case to implement all jobs using just the standard job entries (Start, SQL Scripting, Bulk Loading, Truncate Tables, Success, and so on), if you create a custom Transformation it will result I nthe creation of *ktr file(s) which you will then place into the etl_sales subfolder.

1. Implement the transformations in plain SQL or as Kettle transformations in spoon. Name the transformations or ETL scripts according to the fact/dimension name. Place the resulting files in **`BI_Projects/BI_<groupNo>/task2/etl_sales/`** folder in your solution directory.
2. Create a data integration job that executes the sql scripts and/or transformation(s) in an appropriate sequence and loads all data into the data mart. Name the job "4 – Load BikesDW".
3. Execute your ETL process to populate the data mart (BikesDW) from the staging area database (Bikes).

Note: If you implement your SQL process in plain SQL, use INSERT INTO <table_name> SELECT … commands.

2.3 ROLAP Cube definition

Once the data mart has been populated, you will use Pentaho Schema Workbench to define an OLAP cube, including appropriate hierarchies for rollup and drill-down analyses. In the final part of this exercise (task 3), you will then use the created OLAP cube to formulate Multidimensional Expression Query language (MDX) queries that you will report to management.

1. Start Pentaho Schema Workbench and establish a connection to your OLAP database (Options → Connection…;
2. Create a schema (File → New → Schema) and save it as **BI_Projects/BI_<groupNo>/task2/olap_cube/bike_sales.xml** in your solution directory
3. Add a cube and name it `bike_sales`. Add the following measures:
   a. Quantity (column `OrderQty`)
   b. Revenue (column `OrderLineTotal`)
   c. Profit (column `OrderLineProfit`)
   d. TaxAmount (column `OrderLineTaxAmt`)
   e. ShippingCost (column `OrderLineShippingCost`)

   Choose appropriate aggregators for your measures and "Currency" formatString where appropriate
4. Add the Date dimension
   a. Add the dimension on the top level as a direct subnode of Schema
   b. Set it up as a time dimension (i.e., choose type `TimeDimension`).
   c. Add a single hierarchy named `Days` and set the `primaryKey` field to `DateKey`) and add the Dim_Date table to it.
   d. Add the three temporal levels `Year`, `Month`, and `Day`. For each level, choose the respective table (`CalendarYear`, `MonthNumberOf`, `DayNumberOfMonth`) and column and choose the correct `levelType` (`TimeYears`, `TimeMonths`, `TimeDays`).
5. Define the Location dimension:

   Add Location as a subnode of Schema and add a hierarchy `Territory` with levels `Region` and `Country`. Like in the previous step, set the respective tables, columns and keys.
6. Define the Product dimension:
   a. Add `Product` as a subnode of Schema
   b. Add a hierarchy `ProductCategory` with levels `TopCategory`, and `SubCategory`. Again, set the respective tables, columns and keys.
7. Define the Customer dimension with the following three single-level hierarchies and set the respective tables, columns and keys:
   a. Age
   b. Gender
   c. FullName
8. Add dimension usages to the `bike_sales` cube:
   - `Customer` (with foreign key `CustomerKey`)
   - `Product` (with foreign key ProductKey)
   - `ShippedTo` (using Location with foreign key `ShipToLocationKey`)
   - `OrderDate` (using Date with foreign key `OrderDateKey`)

Make sure you set the foreignKey and source fields of each dimension usage.

Schema Workbench does not warn you about missing definitions (e.g., table, primary key for dimensions). If you don't get any values for your MDX query, check if you have set everything correctly.

Deliverables

In this second task, your **BI_Projects/BI_<groupNo>/task2/** folder now contains the following sub-folders:

- data_mart – data mart table creation scrips;
- etl_sales – sql/ktr files that are used for the ETL process;
- olap_cube – xml file with the cube definition;

Resources

- Mondrian Documentation: http://mondrian.pentaho.com/documentation/index.php
- Youtube playlist: Pentaho Workbench OLAP Cubes – Using Pentaho Schema Workbench http://bit.ly/2gpEcP7
- MySQL Date and Time functions: https://www.w3resource.com/mysql/date-and-time-functions/date-and-time-functions.php

3. Business Analytics (Queries) [12 points]

A number of business users within the Company have contacted your team and requested various custom queries, collected and specified below. Your team is assigned to deliver the requested information using either plain SQL or MDX query language. You can choose which language is more appropriate to answer each of the questions, but you should formulate and **hand in at least three MDX queries** executed in Schema workbench.

Task Description

Implement the following queries using either plain SQL or MDX targeting the data mart or the OLAP cube, respectively. For each query, example result sets are provided. For MDX queries, the (albeit less valuable) default result set format is acceptable. To develop your MDX queries, use File → New → MDX Query in the Schema Workbench.

3.1 Sales – profit for top product categories

What is the profit for each of the top product categories in the year 2021, sorted by profit in descending order? Use `OrderDate` and `OrderLineProfit`.

Product Top Category	Profit
Accessories Bikes ...	355680.00 278390.00 ...

3.2 Sales – revenue for countries

What is the total revenue for each country in the year 2019 sorted by revenue in descending order? Use `ShipToLocation`, `OrderDate` and `OrderLineTotal`.

Country	Revenue
Australia United Kingdom ...	2115624.00 278390.00 ...

3.3 Sales – most profitable customers in the time period

What are the top 10 most profitable customers in the first six months of the year 2021 (January through June) sorted by profit in descending order? Use `OrderDate`, and `OrderLineProfit`.

Customer Rank	Customer Name	Profit
1 2 ... 10	Melody Diaz Sean Brooks ... Samuel V Russell	3200 2950 ... 1700

3.4 Sales – most actively purchasing customers

Who are the top five most active purchasing customers (by the number of purchased items) among the customers in Europe? Use `ShipToLocation`.

Region	Customer Name	Quantity Sold	Rank
Europe	Melinda L Ortega	42	1
Europe	Logan Clark	27	2
...	...	...	...
Europe	Eugene J Sun	13	5

3.5 Logistics - monthly amounts of shipping costs for the customers of a country

What are the total shipping costs for the buyers in the United Kingdom who chose Cargo International as the shipping method in each of the first six months of the year 2020? Use `ShipToLocation`, `OrderDate`, `ShipMethod` and `OrderLineShippingCost`.

Country	Year	Month	Shipping costs
United Kingdom	2020	January	55.10
United Kingdom	2020	February	190.50
...	...	...	...
United Kingdom	2020	June	415.15

3.6 Production – most sold product models in each of the categories

What are the three most sold product models in each of the top product categories, sorted by Product Top Category and Quantity Sold?

Product Top Category	Product Sub Category	Product Model	Quantity Sold
Accessories	Tires and Tubes	Patch kit	2581
Accessories	Bottles and Cages	Water Bottle	2205
Accessories	Tires and Tubes	ML Mountain Tire	2110
Bikes	Touring Bikes	Touring-1000	2544
...	...	...	...

3.7 Accounting – most profitable countries for the goods in the price range

What are the three countries that make the most profit for the Company on the goods in the price range between 1000 and 2000? Use `ShipToLocation`, `UnitPrice` and `OrderLineProfit`.

Country	Profit
USA	386730.50
Germany	255780.88
...	...

3.8 Accounting – taxes charged from the customers in certain countries

What is the monthly amount of taxes for the buyers in France and Germany in the first six months of 2021 (January through June)? Use `OrderDate`, `ShipToLocation` and `OrderLineTaxAmt`.

Calendar Year	Month	Country	Tax Amount
2021	January	France	56730.50
2021	January	Germany	52780.88
2021	February	France	38421.20
2021	February	Germany	42729.65
...	...	...	...

Deliverables

Create a file for each query solution 3_1 through 3_8. With an extension, SQL or MDX. E.g., if the second query was solved with an MDX query, the file should be named 3_2.mdx; or, if it was solved with SQL, it should be named 3_2.sql.

Create a pdf-file with screenshots of the results for each query, name the file **query_results.pdf**

Place all your files from this task into a sub-folder named **task3**.

If necessary, create a separate text file named **comments_XX.txt**.

Resources

- An introduction into MultiDimensional eXpressions (MDX): http://didawiki.di.unipi.it/lib/exe/fetch.php/mds/lbi/ssas2012ch3.pdf

Bonus question [up to 3 points]

The suggested Star schema might reveal itself sub-optimal in certain circumstances.

There are certain deliberately introduced imperfections in this schema.

If you already have an idea about how you could improve this schema, please describe in a few sentences, in a paragraph or two, what might cause issues in this schema and what changes you think can make this schema a little more robust and improve performance. Please place your answer into `comments_XX.txt`

Appendix: Data Dictionary

This is a set of Tab delimited text files comprised of 12 files to be used to perform the lab examples. This data has been extracted from the Company's order transaction system and is ready to be loaded into the staging area. Each of the tables of the staging area is briefly described below. Most of the field names are self-explanatory, but there are some comments in the comments section too. Text files are Tab delimited. Created using Code page 1252 (ANSI – Latin I).

TB_Address

Fields	Key	Type	NULL	Comments
AddressID	PK	INT	NOT	Primary key for Address records
CountryID	FK	INT	NOT	Foreign key constraint referencing TB_Country.CountryID
AddressLine		VARCHAR(60)	NOT	Street address line
City		VARCHAR(30)	NOT	Name of the city
PostalCode		VARCHAR(15)	NOT	Postal code for the street address

TB_Country

Fields	Key	Type	NULL	Comments
CountryID	PK	INT	NOT	Primary key for Country lookup table
Country		VARCHAR(50)	NOT	Country name
Region		VARCHAR(50)	NOT	Region name
TaxRate		DECIMAL(13,4)		Tax rate that depends on the country, used to calculate tax charges.
ShipCoeff		DECIMAL(13,4)		Shipping coefficient that depends on the country, used to calculate shipping costs.

TB_Customer

Fields	Key	Type	NULL	Comments
CustomerID	PK	INT	NOT	Primary key for Customer records
PersonID	FK	INT	NOT	Foreign key constraint referencing TB_Person.PersonID
AccountNumber		VARCHAR(30)	NOT	Customer's account number

TB_OrderStatus

Fields	Key	Type	NULL	Comments
OrderStatusID	PK	INT	NOT	Primary key for Order status lookup table
OrderStatusName		VARCHAR(50)	NOT	Order status name

TB_Person

Fields	Key	Type	NULL	Comments
PersonID	PK	INT	NOT	Primary key for Person records
FirstName		VARCHAR(50)	NOT	
MiddleName		VARCHAR(50)		
LastName		VARCHAR(50)	NOT	
Gender		VARCHAR(1)		‘M’ for ‘Male’, ‘F’ for ‘Female’
Birthdate		DATE		Date of birth

TB_Product

Fields	Key	Type	NULL	Comments
ProductID	PK	INT	NOT	Primary key for Product records
ProductName		VARCHAR(50)	NOT	Product name
ProductNumber		VARCHAR(50)	NOT	Unique product identification number
StandardCost		DECIMAL(13,4)	NOT	Standard cost of production of a product
ListPrice		DECIMAL(13,4)	NOT	List price of a product
ProductSubCategoryID	FK	INT		Foreign key constraint referencing TB_ProductSubCategory.ProductSubCategoryID
ProductModelID	FK	INT		Foreign key constraint referencing TB_ProductModel.ProductModelID
SellStartDate		DATE	NOT	Date the product has become available for sale
SellEndDate		DATE		Date the product is not available for sale anymore

TB_ProductModel

Fields	Key	Type	NULL	Comments
ProductModelID	PK	INT	NOT	Primary key for Product model lookup table
ProductModelName		VARCHAR(50)	NOT	Product model name

TB_ProductSubCategory

Fields	Key	Type	NULL	Comments
ProductSubCategoryID	PK	INT	NOT	Primary key for Product SubCategory lookup table
ProductTopCategoryID	FK	INT	NOT	Foreign key constraint referencing TB_ProductTopCategory. ProductTopCategoryID
Name		VARCHAR(50)	NOT	Product SubCategory name

TB_ProductTopCategory

Fields	Key	Type	NULL	Comments
ProductTopCategoryID	PK	INT	NOT	Primary key for Product TopCategory lookup table
Name		VARCHAR(50)	NOT	Product TopCategory name
ShipSurcharge		DECIMAL(13,4)		Shipping surcharge that depends on the product's TopCategory, used to calculate shipping costs.

TB_SalesOrderDetail

Fields	Key	Type	NULL	Comments
SalesOrderDetailID	PK	INT	NOT	Primary key for Sales Order Details records
SalesOrderID	FK	INT	NOT	Foreign key constraint referencing TB_SalesOrderHeader.SalesOrderID
ProductID	FK	INT	NOT	Foreign key constraint referencing TB_Product.ProductID
OrderQty		INT	NOT	Quantity ordered per product
UnitPrice		DECIMAL(13,4)	NOT	Selling price of a single unit of a product

TB_ShipMethod

Fields	Key	Type	NULL	Comments
ShipMethodID	PK	INT	NOT	Primary key for Shipping methods lookup table
ShipMethodName		VARCHAR(50)	NOT	Shipping method name
ShipBase		DECIMAL(13,4)	NOT	Shipping base that depends on the shipping method, used to calculate shipping costs.
ShipRate		DECIMAL(13,4)	NOT	Shipping rate that depends on the shipping method, used to calculate shipping costs.

TB_SalesOrderHeader

Fields	Key	Type	NULL	Comments
SalesOrderID	PK	INT	NOT	Primary key for Sales Order Details records
SalesOrderNumber		VARCHAR(30)	NOT	Unique sales order identification number.
OrderDate		DATE	NOT	Date the sale order is created.
DueDate		DATE	NOT	Date the order is due to arrive to the customer.
ShipDate		DATE		Date the order is shipped to the customer.
OrderStatusID	FK	INT	NOT	Foreign key constraint referencing TB_OrderStatus.OrderStatusID
CustomerID	FK	INT	NOT	Foreign key constraint referencing TB_Customer.CustomerID
BillToAddressID	FK	INT	NOT	Foreign key constraint referencing TB_Address.AddressID
ShipToAddressID	FK	INT	NOT	Foreign key constraint referencing TB_Address.AddressID
ShipMethodID	FK	INT	NOT	Foreign key constraint referencing TB_ShipMethod.ShipMethodID

Appendix: Folder structure for the assignment deliverable.

```
BI_Projects
  \-BI_XX
    |
    +---csv_data
    |       Address.csv, Product.csv, SalesOrderHeader.csv, ...
    |
    +---task1
    |   \---oltp_db
    |           TB_Address.sql, TB_Product.sql, TB_SalesOrderDetail.sql ...
    |
    +---task2
    |   +---data_mart
    |   |       Dim_Customer.sql, Dim_Product.sql, Fact_InternetSales.sql ...
    |   |
    |   +---etl_sales
    |   |       *.ktr / *.sql
    |   |
    |   \---olap_cube
    |           bike_sales.xml
    |
    +---task3
    |       5_1.mdx / 5_2.mdx / 5_3.mdx ...
    |       5_1.sql / 5_2.sql / 5_3.sql ...
    |   1 - Create Bikes.kjb
    |   2 - Load Bikes.kjb
    |   3 - Create BikesDW.kjb
    |   4 - Load BikesDW.kjb
    |   BI_Bikes_XX.kdb
    |   BI_BikesDW_XX.kdb
    |   comments_XX.txt
    |   group_XX.txt
    |   repository.log
```

* Please make sure to replace **XX** with **Your Group Number** (*.kdb files, also BI_XX folder)