Individual Project Database Foundations

Inventory Model Design Database

Phanindra Panthagani

NetId: pxp180008

##### 

### **Assumptions and Data Cleaning Part +Normalization and Data Modelling**

1. Data Munging:

Each of these tables were then created according to the fields using R and selecting particular fields into particular entities. There were many duplicates for each of the tables and duplicates were removed using Remove Duplicates in Excel. The R code used is as below:

There were many special characters like $,~ etc and NULL Values which were present. The Special Characters were removed and the rows containing the NULL Values were completely removed except in Salary\_History Table.

1. Normalization:

I looked at the data and started Normalizing the data, but I faced a lot of problems while trying to do it. So the first thing I did was to visualize the whole data by preparing the data model and then choosing the appropriate Primary & Foreign Keys.

1. Data Model

The data model looked as below:

A screenshot of a cell phone

Description generated with high confidence

### Salary\_History table Formatting

The Salary\_History table had date in the wrong format by default like mm/dd/yyyy . It was changed to the proper MySQL format which is (YYYY-DD-MM) which was done in excel. For this I used the DATEVALUE function(Julian Date) in Excel and then using Custom Format to change the date to YYYY-DD-MM. A snapshot is shown below.

A screenshot of a cell phone

Description generated with very high confidence

Dealing with NULL VALUES in Salary\_History:

All NULL Values in Salary\_History were replaced with 1999-09-12 this date because MySQL doesn’t import NULL Values and this was used in the Queries. For example in the first Query this date was used for Sal\_End=1999-09-12

Questions:

**What to Do:**

1. Start understanding the data by normalizing the provided datasets using the Data Dictionary. Remember that this takes up to 80% of your time in completing this project, so start early.

**Ans: The dataset was imported in R and then exported into excel after forming the 10 different tables as given in the document. The R code is present at the end of the document named as** splitting into different tables R code

**I looked at the data and started Normalizing the data, but I faced a lot of problems while trying to do it. So the first thing I did was to visualize the whole data by preparing the data model and then choosing the appropriate Primary & Foreign Keys.**

1. Then clean (data munging) each of the corresponding tables, inserting flags, removing bad characters, and filling in missing data fields with known values where appropriate.

**Ans: The data cleaning was done by importing the data in R and then exporting into excel. Foreign Characters like?, ~ etc. were removed and data was cleaned. The R file is attached along with the Report.**

1. Create a Logical and Physical Model using Erwin or DBeaver.

**Ans: Logical & Physical model is as below in the next page:**

A screenshot of a cell phone

Description generated with high confidence

1. Implement in your choice of DBMS (MySQL, MSSQL, or Oracle) (Please remember that Oracle Online has a table size limitation) by importing the cleaned data. Don’t forget to place all required CONSTRAINTS, FOREIGN KEYS, and PRIMARY KEYS.

**Ans: The DBMS used was MySQL. Forward engineering was done after creating the ER diagram using data model. CONSTRAINTS, FOREIGN KEYS, and PRIMARY KEYS were implemented in the data model.**

1. Write SQL Queries to produce the results in answering the below questions. **Include both the SQL Query code and output results in your final report.**
2. Perform a linear regression using R or other preferred statistical application. Interpret the regression analysis to arrive at a suitable forecast.

**Multiple models with AIC & BIC.**

1. Write and submit a report of your findings with all diagrams, graphs, output, and recommendations through eLearning Turnitin by Dec 7, 2018 at 11:59 PM.

Queries:

1. Write a query to display the current salary for each employee in department 300. Assume that only current employees are kept in the system, and therefore the most current salary for each employee is the entry in the salary history with a NULL end date. Sort the output in descending order by salary amount.

**Ans:**

**Query:**

**SELECT S.Employee\_Emp\_Num as EMP\_ID,E.Emp\_Fname as First\_Name,E.Emp\_Lname as Last\_name,S.Sal\_Amount as Current\_Salary**

**FROM salary\_history S INNER JOIN employee E**

**ON S.Employee\_Emp\_Num=E.Emp\_Num**

**WHERE S.Sal\_End = '1999-09-12' AND E.Department\_Dept\_Num=300 ORDER BY Sal\_Amount DESC;**

**Ouptut :**

**A screenshot of a computer

Description generated with very high confidence**

**Since its only 25 rows pasting the Output.**

|  |  |  |  |
| --- | --- | --- | --- |
| EMP\_ID | First\_Name | Last\_name | Current\_Salary |
| 83746 | SEAN | RANKIN | 95550 |
| 84328 | FERN | CARPENTER | 94090 |
| 83716 | HENRY | RIVERA | 85920 |
| 84432 | MERLE | JAMISON | 85360 |
| 83902 | ROCKY | VARGAS | 79540 |
| 83695 | CARROLL | MENDEZ | 79200 |
| 84500 | CHRISTINE | WESTON | 78690 |
| 84594 | ODELL | TIDWELL | 77400 |
| 83910 | LAUREN | AVERY | 76110 |
| 83359 | MERLE | WATTS | 72240 |
| 83790 | LAVINA | ACEVEDO | 72000 |
| 83433 | RONNA | NORWOOD | 68870 |
| 84521 | DELFINA | JUDD | 66000 |
| 83653 | LEEANN | HORN | 61920 |
| 83738 | PORTER | STACY | 58200 |
| 83788 | LANA | DOWDY | 56760 |
| 83867 | TRACIE | KELLY | 56750 |
| 84234 | LUISA | MINER | 54720 |
| 83637 | TANIKA | CRANE | 52870 |
| 83877 | STEPHAINE | DUNLAP | 52650 |
| 84035 | HAL | FISHER | 51600 |
| 83729 | CORRINA | RAMEY | 48500 |
| 83732 | SAMMY | DIGGS | 44720 |
| 83644 | WILLA | MAXWELL | 43200 |
| 83312 | ROSALBA | BAKER | 42400 |

1. Write a query to display the starting salary for each employee. The starting salary would be the entry in the salary history with the oldest salary start date for each employee. Sort the output by employee number.

**Query:**

**SELECT min\_sal.Employee\_Emp\_Num, emp.EMP\_FNAME, emp.EMP\_LNAME, min\_sal.SAL\_FROM, min\_sal.SAL\_AMOUNT FROM**

**(SELECT Employee\_Emp\_Num,Sal\_Amount, MIN(Sal\_FROM) AS SAL\_FROM FROM salary\_history GROUP BY Employee\_Emp\_Num) AS min\_sal INNER JOIN employee emp**

**ON min\_sal.Employee\_Emp\_Num=emp.Emp\_Num**

**ORDER BY min\_sal.Employee\_Emp\_Num;**

**Ouptut:**

**A screenshot of a social media post

Description generated with very high confidence**

**363 rows were returned.They will be attached as Query2Output.CSV**

1. Write a query to display the invoice number, line numbers, product SKUs, product descriptions, and brand ID for sales of sealer and top coat products of the same brand on the same invoice.

**Query:**

**SELECT line.Invoice\_Inv\_Num, line.LINE\_NUM, line.Product\_Prod\_Sku, p1.PROD\_DESCRIPT,p1.Brand\_Brand\_Id**

**FROM line line INNER JOIN(SELECT DISTINCT p.PROD\_SKU,p.PROD\_DESCRIPT,p.Brand\_Brand\_Id FROM product p WHERE p.PROD\_CATEGORY = 'Top Coat') p1**

**ON line.Product\_Prod\_Sku=p1.PROD\_SKU INNER JOIN(SELECT DISTINCT p.PROD\_SKU,p.PROD\_DESCRIPT,p.Brand\_Brand\_Id FROM product p WHERE p.PROD\_CATEGORY = 'Sealer') p2**

**ON line.Product\_Prod\_Sku=p2.PROD\_SKU WHERE p1.Brand\_Brand\_Id=p2.Brand\_Brand\_Id;**

**Output:**

**0 rows were returned. I went back and cross check if there any sealer and top coat products of the same brand on the same invoice.**

1. The Binder Prime Company wants to recognize the employee who sold the most of their products during a specified period. Write a query to display the employee number, employee first name, employee last name, e-mail address, and total units sold for the employee who sold the most Binder Prime brand products between November 1, 2015, and December 5, 2015. If there is a tie for most units sold, sort the output by employee last name.

**Query**

**SELECT emp\_num, First\_name, LASt\_name, email, tot\_q FROM (SELECT SUM(l1.Line\_Qty) AS tot\_q ,emp1.Emp\_Num AS emp\_num, emp1.EMP\_FNAME AS First\_name, emp1.EMP\_LNAME AS LASt\_name, emp1.EMP\_EMAIL AS email FROM line l1 INNER JOIN invoice in1**

**ON l1.Invoice\_Inv\_Num=in1.Inv\_Num INNER JOIN employee emp1**

**ON in1.Employee\_Emp\_Num=emp1.Emp\_Num INNER JOIN product p1**

**ON l1.Product\_Prod\_Sku=p1.PROD\_SKU INNER JOIN brAND b1**

**ON b1.Brand\_Id=p1.Brand\_Brand\_Id WHERE b1.BRAND\_NAME = 'BINDER PRIME' AND in1.Inv\_Num not like '-%'**

**GROUP BY emp1.Emp\_Num) q1 WHERE tot\_q = (SELECT MAX(abc1.tot\_q) FROM (SELECT SUM(l1.Line\_Qty) AS tot\_q ,emp1.Emp\_Num AS emp\_num,**

**emp1.EMP\_FNAME AS First\_name, emp1.EMP\_LNAME AS LASt\_name, emp1.EMP\_EMAIL AS email FROM line l1 INNER JOIN invoice in1**

**ON l1.Invoice\_Inv\_Num=in1.Inv\_Num INNER JOIN employee emp1**

**ON in1.Employee\_Emp\_Num=emp1.Emp\_Num INNER JOIN product p1 ON l1.Product\_Prod\_Sku=p1.PROD\_SKU INNER JOIN brAND b1**

**ON b1.Brand\_Id=p1.Brand\_Brand\_Id**

**WHERE b1.BRAND\_NAME = 'BINDER PRIME' AND in1.Inv\_Num not like '-%'**

**AND in1.INV\_DATE between ('2015-11-01') AND ('2015-12-05') GROUP BY emp1.Emp\_Num) abc1)**

**ORDER BY LASt\_name;**

**Output:**

**0 rows**

**We see from INVOICE table that there is no invoice for the year 2015 and that’s why we are getting empty output.**

**To check if the Query is right, I removed the inv.INV\_DATE between part and checked if the query is returning any rows. I could get rows returned when I Removed the INV\_DATE constraint , I got 2 rows as Output as below:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **emp\_num** | **First\_name** | **LASt\_name** | **email** | **tot\_q** |
| **84078** | **DIEGO** | **ERWIN** | **E.DIEGO98@LGCOMPANY.COM** | **27** |
| **84106** | **FELICE** | **SAMUEL** | **S.FELICE98@LGCOMPANY.COM** | **27** |

1. Write a query to display the customer code, first name, and last name of all customers who have had at least one invoice completed by employee 83649 and at least one invoice completed by employee 83677. Sort the output by customer last name and then first name.

**Query:**

**SELECT a.Cust\_Code, a.cust\_lname, a.cust\_fname FROM (SELECT c.Cust\_Code, c.cust\_fname, c.cust\_lname FROM customer c INNER JOIN**

**invoice i ON c.Cust\_Code = i.Customer\_Cust\_Code WHERE i.Employee\_Emp\_Num = 83649) a INNER JOIN**

**(SELECT c.Cust\_Code, c.cust\_fname,c.cust\_lname**

**FROM customer c INNER JOIN invoice i ON c.Cust\_Code = i.Customer\_Cust\_Code WHERE i.Employee\_Emp\_Num = 83677) b**

**ON a.Cust\_Code=b.Cust\_Code ORDER BY a.cust\_lname, a.cust\_fname;**

**Ouput:**

**0 rows returned.**

1. LargeCo is planning a new promotion in Alabama (AL) and wants to know about the largest purchases made by customers in that state. Write a query to display the customer code, customer first name, last name, full address, invoice date, and invoice total of the largest purchase made by each customer in Alabama. Be certain to include any customers in Alabama who have never made a purchase (their invoice dates should be NULL and the invoice totals should display as 0).

**Query:**

**SELECT cus.Cust\_Code, cus.cust\_fname, cus.cust\_lname, cus.cust\_street, cus.cust\_city, cus.cust\_state, cus.cust\_zip, inv1.inv\_date,**

**MAX(COALESCE(inv1.Invoice\_Total,0)) AS MAX\_inv FROM customer cus left outer JOIN invoice inv1 ON**

**cus.Cust\_Code = inv1.Customer\_Cust\_Code WHERE cus.cust\_state = 'AL' AND inv1.Inv\_Num not like '-%' GROUP BY cus.Cust\_Code;**

**Output:**

**A screenshot of a computer

Description generated with very high confidence**

**50 rows were returned, and output is attached in OutputQuery6.csv**

1. One of the purchasing managers is interested in the impact of product prices on the sale of products of each brand. Write a query to display the brand name, brand type, average price of products of each brand, and total units sold of products of each brand. Even if a product has been sold more than once, its price should only be included once in the calculation of the average price. However, you must be careful because multiple products of the same brand can have the same price, and each of those products must be included in the calculation of the brand’s average price.

**Query:**

**SELECT BRAND\_NAME, BRAND\_TYPE, AVGprice, UnitsSold FROM brAND b JOIN (SELECT p.Brand\_Brand\_id, AVG(p.prod\_price) AS AVGprice**

**FROM product p GROUP BY Brand\_Brand\_Id) sub1**

**ON b.Brand\_Id = sub1.Brand\_Brand\_Id JOIN (SELECT p.Brand\_Brand\_id, SUM(l.Line\_Qty) AS UnitsSold FROM product p JOIN line l**

**ON p.Prod\_Sku = l.Product\_Prod\_Sku GROUP BY Brand\_Brand\_Id) sub2**

**ON b.Brand\_Id = sub2.Brand\_Brand\_Id ORDER BY BRAND\_NAME;**

**Output:**

**A screenshot of a computer

Description generated with very high confidence**

**9 rows were returned as attached.**

1. The purchasing manager is still concerned about the impact of price on sales. Write a query to display the brand name, brand type, product SKU, product description, and price of any products that are not a premium brand, but that cost more than the most expensive premium brand products.

**Query:**

**SELECT brd.BRAND\_NAME, brd.BRAND\_TYPE, prd.prod\_sku, prd.prod\_descript, prd.prod\_price FROM product prd INNER JOIN brand brd**

**ON prd.Brand\_Brand\_Id=brd.Brand\_Id WHERE brd.BRAND\_TYPE <> 'PREMIUM' AND prd.prod\_price > (SELECT MAX(prd1.prod\_price)**

**FROM product prd1 INNER JOIN BRAND brd1**

**ON prd1.Brand\_Brand\_Id=brd1.Brand\_Id**

**WHERE brd1.BRAND\_TYPE = 'PREMIUM');**

**Ouptut:**

A screenshot of a social media post

Description generated with very high confidence

**There is only 1 such brand name, brand type, product SKU, product description, and price of any products that are not a premium brand, but that cost more than the most expensive premium brand products**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| BRAND\_NAME | BRAND\_TYPE | prod\_sku | prod\_descript | prod\_price |
| LONG HAUL | CONTRACTOR | 1964-OUT | Fire Resistant Top Coat, for Interior Wood | 78.49 |

1. Using SQL descriptive statistics functions calculate the value of the following items:
   1. What are the products that have a price greater than $50?

**Query:**

**SELECT \* FROM product WHERE prod\_price > 50;**

**Output:**

**A screenshot of a computer

Description generated with very high confidence**

**There are 3 products with price greater than $50 as below**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Prod\_Sku | Prod\_Descript | Prod\_Type | Prod\_Base | Prod\_Category | Prod\_Price | Prod\_QOH | Prod\_Min | Brand\_Brand\_Id |
| 1021-MTI | Elastomeric, Exterior, Industrial Grade, Water Based | Exterior | Water | Top Coat | 62.99 | 22 | 25 | 35 |
| 1964-OUT | Fire Resistant Top Coat, for Interior Wood | Interior | Solvent | Top Coat | 78.49 | 120 | 10 | 30 |
| 3694-XFJ | Epoxy-Modified Latex, Interior, Semi-Gloss (MPI Gloss Level 5) | Interior | Water | Top Coat | 54.89 | 39 | 25 | 27 |

* 1. What is total value of our entire inventory on hand?

**Query:**

**SELECT SUM(prod\_qoh\*prod\_price) AS 'Total\_Value' FROM product;**

**Output:**

A screenshot of a computer

Description generated with very high confidence

**The total value of the inventory is : $360307.7926878929**

* 1. How many customers do we presently have and what is the total of all customer balances?

**Query:**

**SELECT count(distinct(c.Cust\_Code)) AS "NumberofCustomers", SUM(c.cust\_balance) AS "TotalofAll\_Customer\_Balances"**

**FROM customer c;**

**Output:**

**A screenshot of a computer

Description generated with very high confidence**

**There are a total of 1362 customers and the**

**Total of all Customer Balances is $787201.150834322**

* 1. What are to top three states that buy the most product in dollars from the company?

**Query:**

**SELECT cus.CUST\_STATE,SUM(inv.Invoice\_Total) AS inv\_tot FROM customer cus INNER JOIN invoice inv**

**ON cus.Cust\_Code=inv.Customer\_Cust\_Code WHERE inv.Inv\_Num not like '-%'**

**GROUP BY cus.CUST\_STATE**

**ORDER BY inv\_tot desc limit 3;**

**Output:**

**A screenshot of a social media post

Description generated with very high confidence**

**The top 3 states that buy the most product in dollars from the company are PA,NY and NC with the invoice totals as below:**

|  |  |
| --- | --- |
| **CUST\_STATE** | **inv\_tot** |
| **PA** | **38618.12** |
| **NY** | **32242.93** |
| **NC** | **19611.39** |

1. Using predictive statistics calculate what the predicted forecast of sales for the next year based on the INV\_DATE (independent) and INV\_TOTAL (dependent). Remember that you will need to convert the INV\_DATE from the MS SQL Server stored date value to the expect Julian date, since numbers in MS SQL are stored as the number of days since 1/1/1900 with the fraction as the portion of a day (if you are using a different DBMS use the appropriate code for conversion.)

declare @d1 datetime

set @d1 = 41867

select @d1

select CONVERT(varchar(20),@d1,120)

or if you want to do it in one statement:

select CONVERT(varchar(25),cast(41867 as datetime),120)

Analyze your results from the linear regression, and provide the R2, model, coefficients, and the confidence interval for your analysis.

**Ans:**

I converted the DATETIME values to Julian Date Time and then used R to evaluate various models.

The R Code gave very bad results as our data is time series data.

Hence, I converted the whole data into months and used the same for prediction. Please find the model summary and Results below

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SUMMARY OUTPUT REGRESSION |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| *Regression Statistics* | |  |  |  |  |  |  |  |
| Multiple R | 0.03802 |  |  |  |  |  |  |  |
| R Square | 0.00145 |  |  |  |  |  |  |  |
| Adjusted R Square | 0.0007 |  |  |  |  |  |  |  |
| Standard Error | 128.892 |  |  |  |  |  |  |  |
| Observations | 1346 |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
| ANOVA |  |  |  |  |  |  |  |  |
|  | *df* | *SS* | *MS* | *F* | *Significance F* |  |  |  |
| Regression | 1 | 32322.66263 | 32322.66263 | 1.945597976 | 0.163292475 |  |  |  |
| Residual | 1344 | 22328178.34 | 16613.22793 |  |  |  |  |  |
| Total | 1345 | 22360501 |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |
|  | *Coefficients* | *Standard Error* | *t Stat* | *P-value* | *Lower 95%* | *Upper 95%* | *Lower 95.0%* | *Upper 95.0%* |
| Intercept | -2567.4 | 1986.876061 | -1.292174585 | 0.196518821 | -6465.106377 | 1330.324879 | -6465.11 | 1330.325 |
| 41317 | 0.06669 | 0.047813537 | 1.394846936 | 0.163292475 | -0.027104715 | 0.160489846 | -0.0271 | 0.16049 |

**2015 Forecasting Invoice Total:**

The given data is a time-series , hence I used excel to use time-series to forecast of next year sales based on the monthly sum total of sales and dividing into monthly chunks.

Model Summary for R

|  |  |  |  |
| --- | --- | --- | --- |
| Model Name | R^2 | Alkaline Information Criteria(AIC) | Bayesian Information Criteria(BIC) |
| Model\_Linear | 0.01357 | 17099.97 | 17115.62 |
| Model\_Quadratic | 0.01356 | 17099.98 | 17115.63 |
| Model\_Cubic | 0.0006201 | 17099.98 | 17115.63 |
| Model\_Differencing Time Series) | 4.877e-05 | 17995 | 18010.65 |

From the above table, We see that linear model has the highest R^2 and hence is the best model. We have the least AIC,BIC values from which we select the best model with the least values. Here the linear model has the AIC BIC values with 17099.97,17115.62.

Summary of model\_linear

A screenshot of a social media post

Description generated with very high confidence

Summary of model\_Quadratic

A screenshot of a cell phone

Description generated with very high confidence

Summary of model\_cubic

**A screenshot of a social media post

Description generated with very high confidence**

Summary of model\_differencingA screenshot of a social media post

Description generated with very high confidence

R Code for Prediction:

Question10DB

Phanindra

December 7, 2018

## R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see <http://rmarkdown.rstudio.com>.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

library(data.table)  
library(DBI)  
library(RSQLite)  
library(tidyverse)

## -- Attaching packages ------------------------------ tidyverse 1.2.1 --

## v ggplot2 3.1.0 v purrr 0.2.5  
## v tibble 1.4.2 v dplyr 0.7.8  
## v tidyr 0.8.2 v stringr 1.3.1  
## v readr 1.1.1 v forcats 0.3.0

## -- Conflicts --------------------------------- tidyverse\_conflicts() --  
## x dplyr::between() masks data.table::between()  
## x dplyr::filter() masks stats::filter()  
## x dplyr::first() masks data.table::first()  
## x dplyr::lag() masks stats::lag()  
## x dplyr::last() masks data.table::last()  
## x purrr::transpose() masks data.table::transpose()

library(ggplot2)  
library(lmtest)

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

library(sandwich)  
library(margins)  
library(partykit)

## Loading required package: grid

## Loading required package: libcoin

## Loading required package: mvtnorm

library(plm)

## Loading required package: Formula

##   
## Attaching package: 'plm'

## The following objects are masked from 'package:dplyr':  
##   
## between, lag, lead

## The following object is masked from 'package:data.table':  
##   
## between

library(dplyr)  
  
  
Invoice <- fread("Invoice.csv")   
View(Invoice)  
inv\_date <- Invoice$`JULIAN INV\_DATE`  
class(inv\_date)

## [1] "integer"

inv\_date <- as.numeric( Invoice$`JULIAN INV\_DATE`)  
class(inv\_date)

## [1] "numeric"

inv\_total <- Invoice$INV\_TOTAL  
class(inv\_total)

## [1] "numeric"

Model\_Linear<- lm(inv\_total~ inv\_date, data=Invoice)  
summary(Model\_Linear)#Multiple R-squared: 0.001357, Adjusted R-squared: 0.0006231

##   
## Call:  
## lm(formula = inv\_total ~ inv\_date, data = Invoice)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -203.46 -102.57 -13.67 77.55 488.89   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -2.475e+03 1.970e+03 -1.256 0.209  
## inv\_date 6.445e-02 4.741e-02 1.360 0.174  
##   
## Residual standard error: 128.7 on 1360 degrees of freedom  
## Multiple R-squared: 0.001357, Adjusted R-squared: 0.0006231   
## F-statistic: 1.849 on 1 and 1360 DF, p-value: 0.1742

AIC(Model\_Linear)#17099.97

## [1] 17099.97

BIC(Model\_Linear)#17115.62

## [1] 17115.62

Model\_Quadratic <- lm(inv\_total~ I(inv\_date^2), data=Invoice)  
summary(Model\_Quadratic)#Multiple R-squared: 0.001356, Adjusted R-squared: 0.0006216

##   
## Call:  
## lm(formula = inv\_total ~ I(inv\_date^2), data = Invoice)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -203.46 -102.58 -13.66 77.54 488.90   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -1.136e+03 9.856e+02 -1.153 0.249  
## I(inv\_date^2) 7.756e-07 5.707e-07 1.359 0.174  
##   
## Residual standard error: 128.7 on 1360 degrees of freedom  
## Multiple R-squared: 0.001356, Adjusted R-squared: 0.0006216   
## F-statistic: 1.846 on 1 and 1360 DF, p-value: 0.1744

AIC(Model\_Quadratic)#17099.98

## [1] 17099.98

BIC(Model\_Quadratic)#17115.63

## [1] 17115.63

Model\_Cubic <- lm(inv\_total~ I(inv\_date^3), data=Invoice)   
summary(Model\_Cubic)#Multiple R-squared: 0.001354, Adjusted R-squared: 0.0006201

##   
## Call:  
## lm(formula = inv\_total ~ I(inv\_date^3), data = Invoice)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -203.46 -102.59 -13.66 77.54 488.90   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -6.897e+02 6.574e+02 -1.049 0.294  
## I(inv\_date^3) 1.244e-11 9.162e-12 1.358 0.175  
##   
## Residual standard error: 128.7 on 1360 degrees of freedom  
## Multiple R-squared: 0.001354, Adjusted R-squared: 0.0006201   
## F-statistic: 1.844 on 1 and 1360 DF, p-value: 0.1747

AIC(Model\_Cubic)#17099.98

## [1] 17099.98

BIC(Model\_Cubic)#17115.63

## [1] 17115.63

Model\_Differencing <- lm(diff(inv\_total)~ diff(inv\_date), data=Invoice)   
summary(Model\_Differencing)#Multiple R-squared: 4.877e-05, Adjusted R-squared: -0.000687

##   
## Call:  
## lm(formula = diff(inv\_total) ~ diff(inv\_date), data = Invoice)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -616.06 -117.79 0.88 120.52 544.83   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) -0.006656 4.867366 -0.001 0.999  
## diff(inv\_date) -0.015095 0.058632 -0.257 0.797  
##   
## Residual standard error: 179.6 on 1359 degrees of freedom  
## Multiple R-squared: 4.877e-05, Adjusted R-squared: -0.000687   
## F-statistic: 0.06628 on 1 and 1359 DF, p-value: 0.7969

AIC(Model\_Differencing)#17995

## [1] 17995

BIC(Model\_Differencing)#18010.65

## [1] 18010.65

## The best model based on prediction came out to be Linear model. Using it for predicting the values for 2015.

#The Julian date for 2/12/2013 based on the previous invoice is 41317. Julian date is the number of days since 1/1/1990.   
#So for 1/1/2015 Julian Date=41317+(Days difference between 2/12/2013 and 1/1/1990)   
Juliandate2015firstday <- 41317+(365-(28+31))+365 #41988  
Juliandate2015lastday <- Juliandate2015firstday+365  
JulianVector2015 <- seq(from=Juliandate2015firstday,to=Juliandate2015lastday,by=1)  
  
#Predicting values for 2015  
InvoiceTotal2015 <- predict(Model\_Linear,data=JulianVector2015)  
  
  
PredictedValuesfor2015=list("JulianDate2015"=JulianVector2015,"Predicted2015INVTOTAL"=InvoiceTotal2015)

splitting into different tables R code :

RDatabaseBUAN6320

Phanindra

November 28, 2018

dataset2<- read.delim("BUAN6320-DataSet2.txt",header=TRUE,sep="\t",  
 dec = ".", fill = TRUE, comment.char = "")  
dataset3<-read.delim("BUAN6320-DataSet3.txt",header=TRUE,sep="\t",  
 dec = ".", fill = TRUE, comment.char = "")  
dataset4<-read.delim("BUAN6320-DataSet4.txt",header=TRUE,sep="\t",  
 dec = ".", fill = TRUE, comment.char = "")  
dataset2omitted <- na.omit(dataset2)  
dataset3omitted <- na.omit(dataset3)  
dataset4omitted <- na.omit(dataset4)  
#Dataset3 has columns X,X.1,X.2 which have all NA values, SO they have to be removed.  
dataset3removedX <- dataset3[,-c(18,19,20)]  
dataset3omitted <- na.omit(dataset3removedX)  
#Now dataset3 has all the 252 observations which were present initially.  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

vendor <- dataset3omitted %>%   
 dplyr::select(VEND\_STREET,VEND\_NAME,VEND\_STATE,VEND\_CITY,VEND\_ID,VEND\_ZIP)  
#Now exporting the vendor file as a CSV file  
write.csv(vendor,file="vendor.csv")  
  
#Now finding the customer table as designed   
customer <- dataset2omitted %>%   
 dplyr::select(CUST\_CODE,CUST\_FNAME,CUST\_LNAME,CUST\_STREET,CUST\_CITY,CUST\_STATE,CUST\_ZIP,CUST\_BALANCE)  
write.csv(customer,file="customer.csv")  
  
#Getting the data for brand  
brand <- dataset3omitted %>%   
 dplyr::select(BRAND\_NAME,BRAND\_TYPE,BRAND\_ID)  
write.csv(brand,file="brand.csv")  
#Getting data for product   
product <- dataset3omitted %>%   
 dplyr::select(PROD\_SKU,PROD\_DESCRIPT,PROD\_TYPE,PROD\_BASE,PROD\_CATEGORY,PROD\_PRICE,PROD\_QOH,PROD\_MIN,BRAND\_ID)  
write.csv(product,file="product.csv")  
  
##Getting invoice  
invoice <- dataset2omitted %>%   
 dplyr::select(INV\_NUM,INV\_DATE,INV\_TOTAL,EMPLOYEE\_ID,CUST\_CODE)  
write.csv(invoice,file="invoice.csv")  
  
  
  
  
  
##After getting all the csv files, duplicates were removed in excel using remove duplicates in Data tab.  
#Getting data for department  
Department <- dataset4omitted %>%   
 dplyr :: select ( DEPT\_NUM, DEPT\_NAME, DEPT\_MAIL\_BOX, DEPT\_PHONE, SUPV\_EMP\_NUM)  
write.csv(Department, file= "Department.csv")  
  
##Getting salary history   
Salary\_history <- dataset4omitted %>%   
 dplyr :: select (SAL\_FROM,SAL\_END,SAL\_AMOUNT,EMP\_NUM)  
write.csv(Salary\_history, file= "Salary\_history.csv")  
  
##Getting line   
line <- dataset2omitted %>%   
 dplyr :: select (LINE\_NUM,LINE\_QTY,LINE\_PRICE,INV\_NUM,PROD\_SKU)  
write.csv(line, file= "line.csv")  
  
##Supplies   
Supplies <- dataset3omitted %>%   
 dplyr :: select (PROD\_SKU,VEND\_ID)  
write.csv(Supplies,file="supplies.csv")