Indiana University Bloomington

CSCI B 565

Data Mining

Data Analytics for IU Bus System

Group Name : RedMiners

Authors:
Jayasankar, Siddharth
Nagarajan, Ganesh
Madhavan, Sarvothaman

Supervisor:
Dr. Dalkilic, Mehmet

December 18, 2015

Abstract

University Bus systems are widely used for shuttling students, faculty and patrons in and out of University campuses and still remains the easiest way to reach the University Campuses. Indiana University operates its bus system under the name IU Bus and the details are available at `http://www.iubus.indiana.edu/campus_bus/index.html`

This paper attempts to develop an uniform framework and processes for analyzing IU Bus data and thus along the process hopes to establish a general framework. Any such framework should be able to transform practical nuances into presentable data analytics. The areas focused in this paper consider various aspects of discussion with the key stake holders, developing glossary, developing metrics for measuring performance,data pipe lining, data visualization, Model creation and model verification.

This project uses MySql as its preferred database for storing operational data, Neo4j graphical database for holding aggregated data, Tableau Public for visualizing the data and R for model creation and verification.

Contents

1 Introduction

University Bus systems are critical for shuttling students in and out of university campuses. These bus systems are not only used at Indiana University, but also at other university systems like University Transit Service,University of Virginia `http://www.virginia.edu/parking/uts/`, The Bus, Boston University system `http://www.bu.edu/thebus/`. All these universities have similar kind of schedules for different terms like Spring, Fall and special trips during late nights. All these services have an live tracking system which records the bus statuses, Geo-locations of buses and their stops and is presented real-time to the users.

IU Bus system has four routes A,B,E and X and has about 52 stops including the minor and major stops. These busses are scheduled about one bus in every 15 minutes for use of its passengers. IU Bus System commits itself to keep up the time as per the schedule at `http://www.iubus.indiana.edu/campus_bus/bus_schedule.html`, however as with any transportation system, the bus gains or looses speed during the course of the travel.

Thus, primary interest of this study is to understand the time difference between the schedule time and the actual time, $t_{schedule} - t_{actual}$. An model was created in Neo4j depicting the acutal bus system, and the ability of the graph database serves in the shortest paths problem and going from place A to place B.
In addition to the above said studies, an statistical model was developed in R to predict the arrival status of the bus at a given stop. The statuses can be On-Time, Delayed and Early, an detailed explanation is given in the corresponding sections. Other than this, a reporting solution was developed in Tableau and is hosted on cloud at location given below. Here, detailed reporting of status of each trips for Spring 2015 for Route A is available. This includes whether the bus arrival status and the seconds by which the bus is early / on-time.

Following are the locations of the project artifacts:

1. Database Schemas
    (a) Hostname : rdc04.uits.iu.edu

   (b) Port : 3099
   (c) Username : bususer
   (d) Password : Password123

2. Neo4j Cloud Database : `http://iubus.sb02.stations.graphenedb.com:24789/browser/`

3. Tableau Public : `https://public.tableau.com/profile/ganesh.nagarajan#!/vizhome/RouteA/IUBus`

4. GitHub Project URL : `https://github.com/sarvothaman/nefarious-octo-rutabaga`

All the codes, data extracts, models, templates related to the project are available at the Github location specified above.

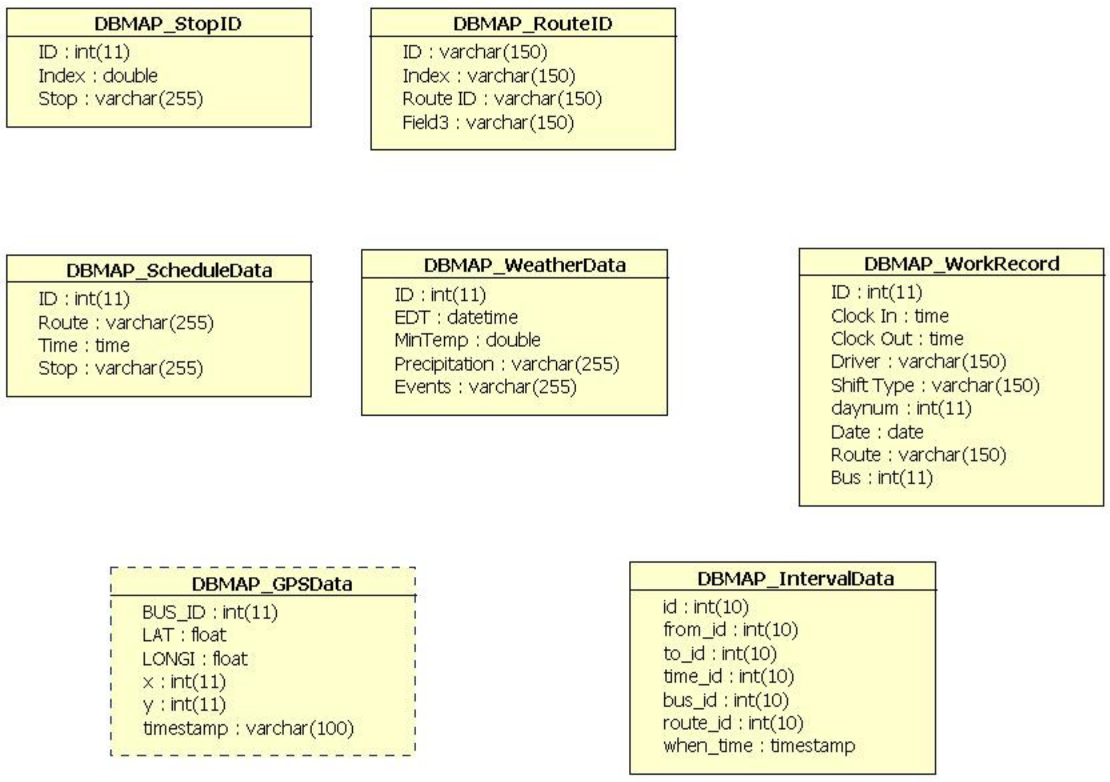
2 Problem Description

3 Constraints

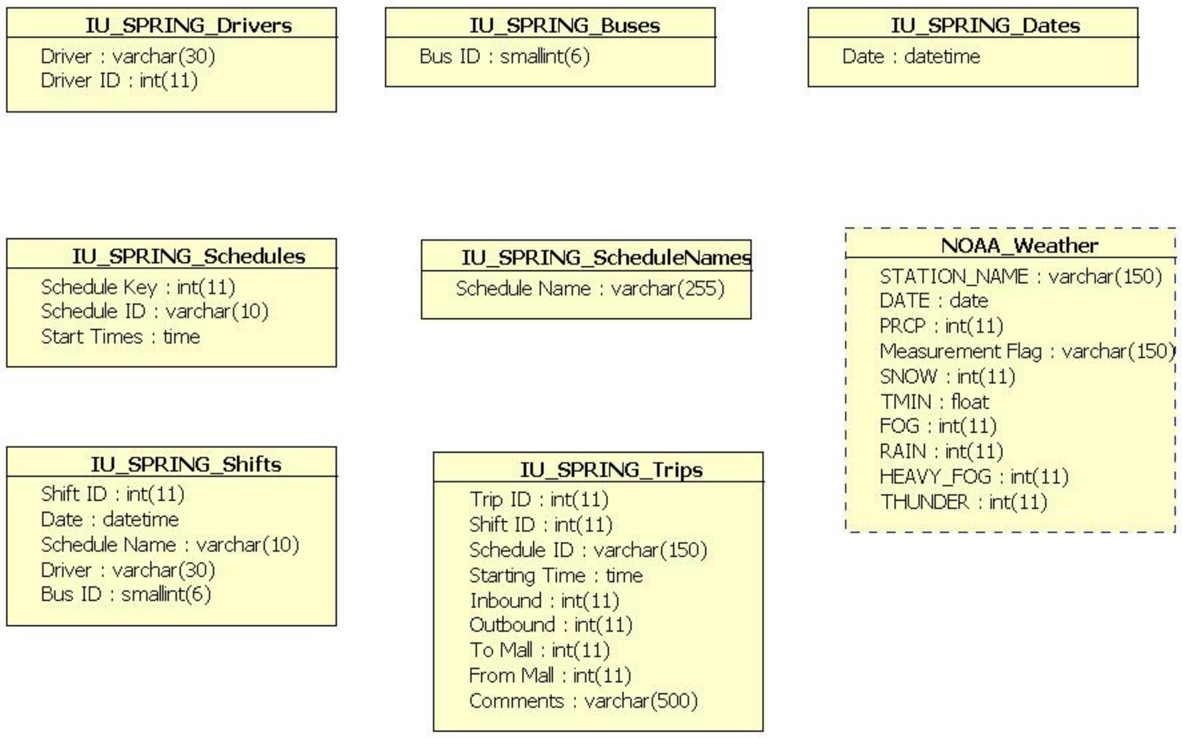
4 Data Management

4.1 Source Data Model

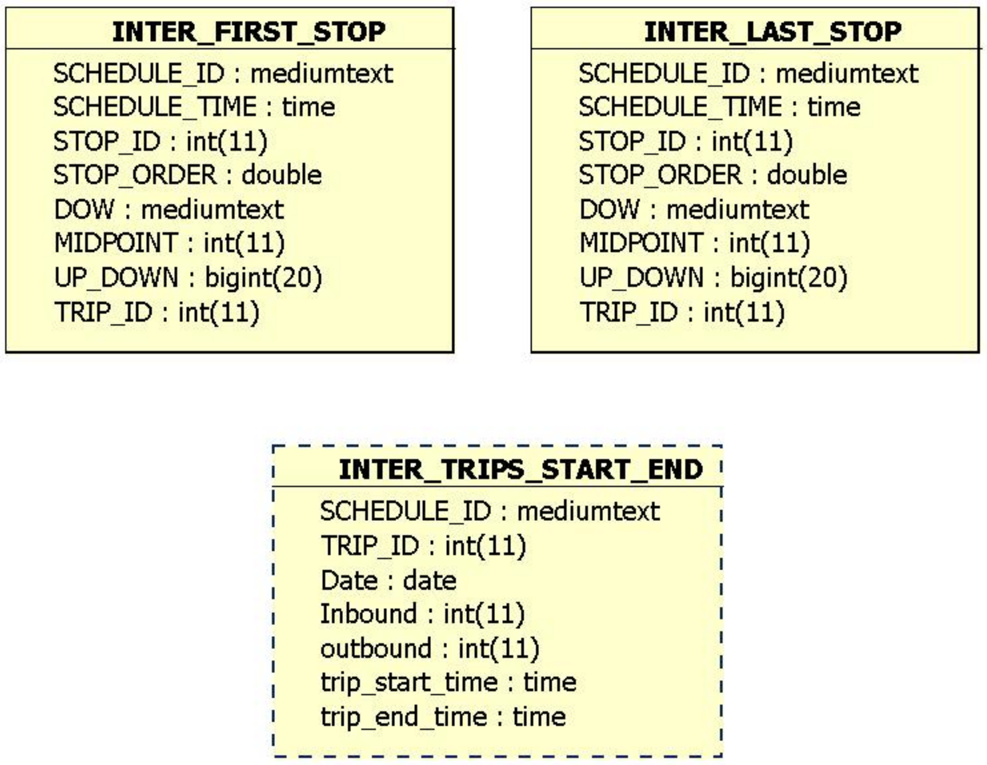
Text Explanation here



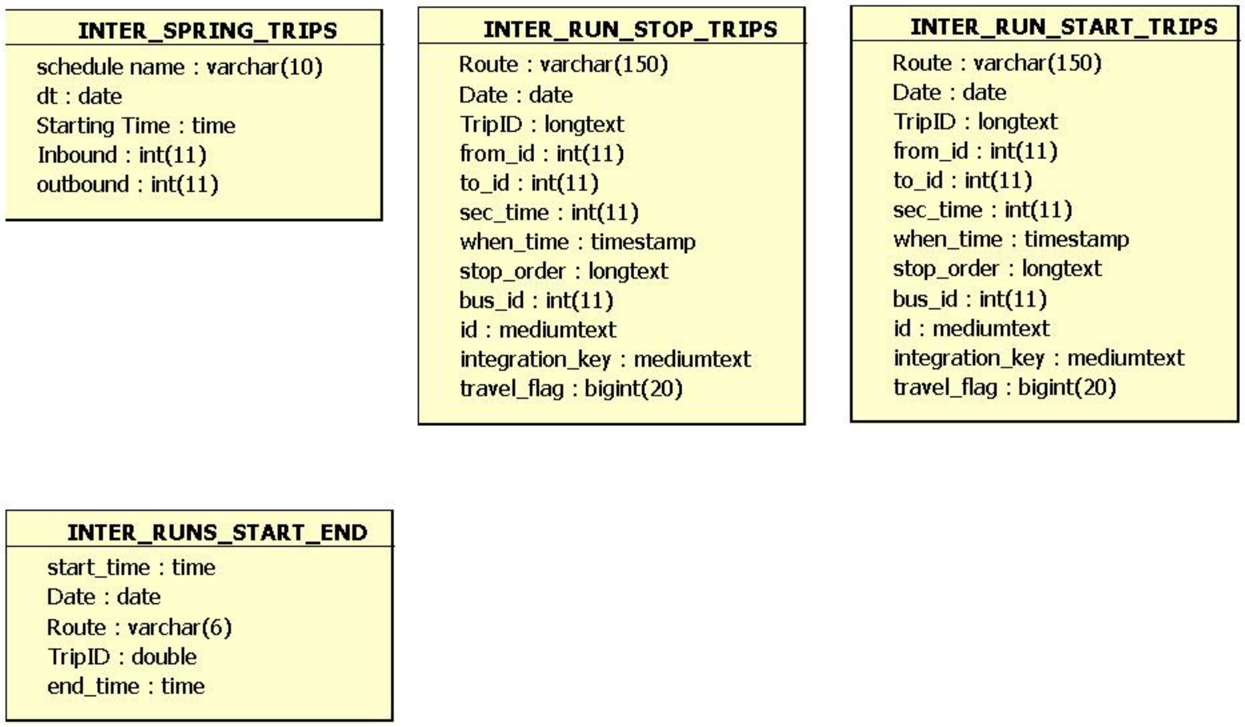
Text Explanation here


IU_SPRING_Drivers
Driver : varchar(30)
Driver ID : int(11)
IU_SPRING_Buses
Bus ID : smallint(6)
IU_SPRING_Dates
Date : datetime
IU_SPRING_Schedules
Schedule Key : int(11)
Schedule ID : varchar(10)
Start Times : time
IU_SPRING_ScheduleNames
Schedule Name : varchar(255)
NOAA_Weather
STATION_NAME : varchar(150)
DATE : date
PRCP : int(11)
Measurement Flag : varchar(150)
SNOW : int(11)
TMIN : float
FOG : int(11)
RAIN : int(11)
HEAVY_FOG : int(11)
THUNDER : int(11)
IU_SPRING_Shifts
Shift ID : int(11)
Date : datetime
Schedule Name : varchar(10)
Driver : varchar(30)
Bus ID : smallint(6)
IU_SPRING_Trips
Trip ID : int(11)
Shift ID : int(11)
Schedule ID : varchar(150)
Starting Time : time
Inbound : int(11)
Outbound : int(11)
To Mall : int(11)
From Mall : int(11)
Comments : varchar(500)

4.2 Intermediate Model



Text Explanation here



INTER_SPRING_TRIPS

schedule name : varchar(10)
dt : date
Starting Time : time
Inbound : int(11)
outbound : int(11)

INTER_RUN_STOP_TRIPS

Route : varchar(150)
Date : date
TripID : longtext
from_id : int(11)
to_id : int(11)
sec_time : int(11)
when_time : timestamp
stop_order : longtext
bus_id : int(11)
id : mediumtext
integration_key : mediumtext
travel_flag : bigint(20)

INTER_RUN_START_TRIPS

Route : varchar(150)
Date : date
TripID : longtext
from_id : int(11)
to_id : int(11)
sec_time : int(11)
when_time : timestamp
stop_order : longtext
bus_id : int(11)
id : mediumtext
integration_key : mediumtext
travel_flag : bigint(20)

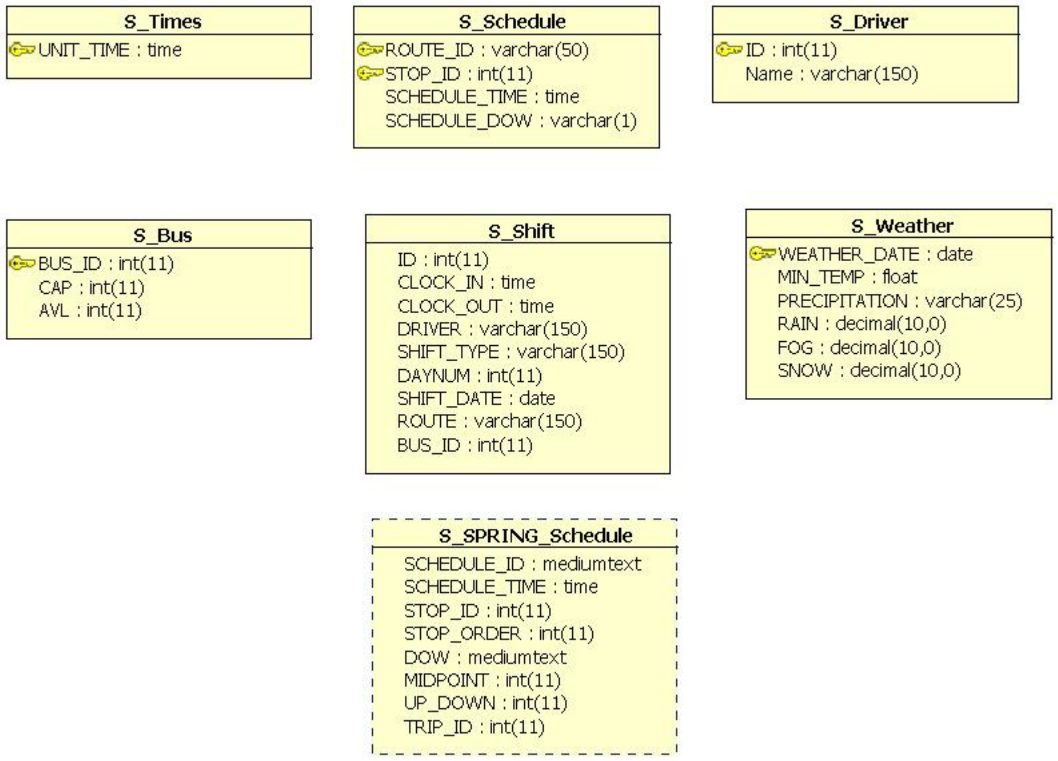
INTER_RUNS_START_END

start_time : time
Date : date
Route : varchar(6)
TripID : double
end_time : time

Text Explanation here

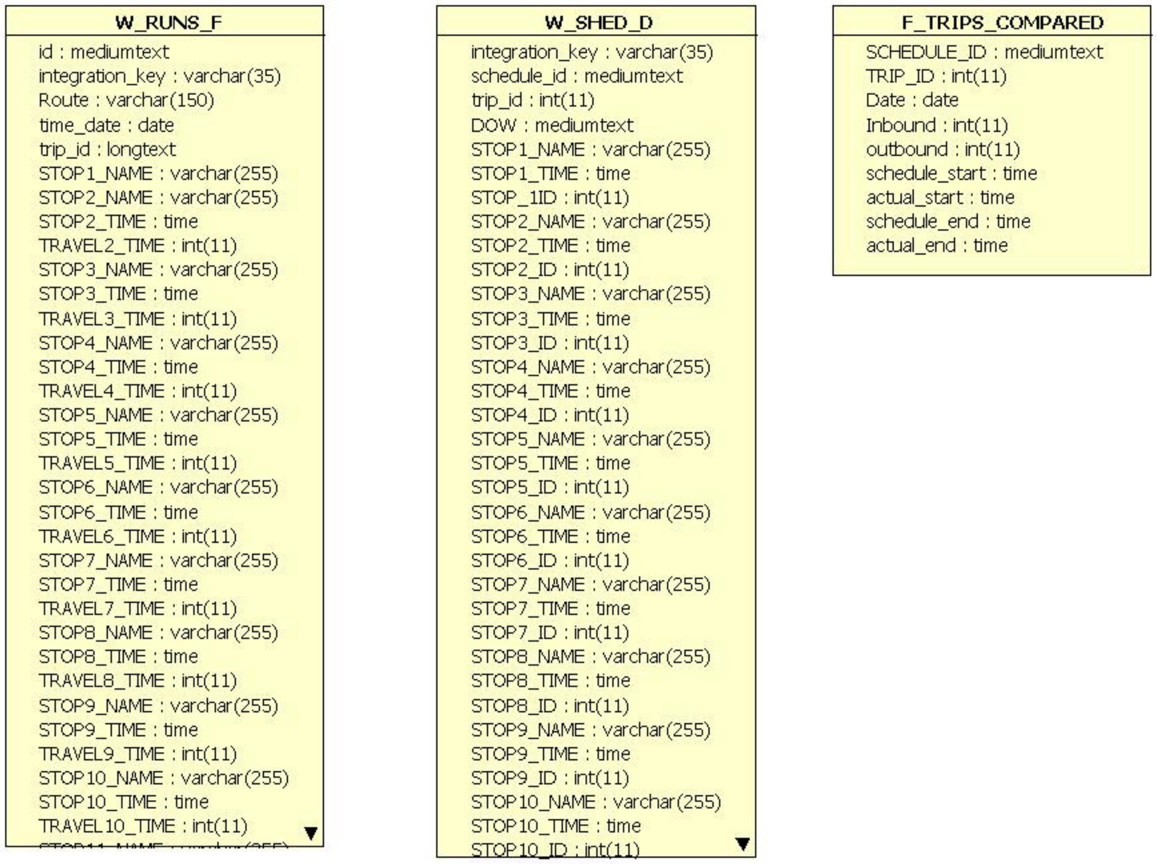
4.3 Final Data Model

Text Explanation here



Data Warehouse Model

The data model by far developed is an OLTP model, which is used for operational efficiency. Inorder to facilitate a better analytic needs, these abstract entities are to be converted to a dimension and fact model. However, instead of traditional OLAP model, we have considered that the dimension to be the schedule and run to be facts. They reside in the model as shown below.



W_RUNS_F
id : mediumtext
integration_key : varchar(35)
Route : varchar(150)
time_date : date
trip_id : longtext
STOP1_NAME : varchar(255)
STOP2_NAME : varchar(255)
STOP2_TIME : time
TRAVEL2_TIME : int(11)
STOP3_NAME : varchar(255)
STOP3_TIME : time
TRAVEL3_TIME : int(11)
STOP4_NAME : varchar(255)
STOP4_TIME : time
TRAVEL4_TIME : int(11)
STOP5_NAME : varchar(255)
STOP5_TIME : time
TRAVEL5_TIME : int(11)
STOP6_NAME : varchar(255)
STOP6_TIME : time
TRAVEL6_TIME : int(11)
STOP7_NAME : varchar(255)
STOP7_TIME : time
TRAVEL7_TIME : int(11)
STOP8_NAME : varchar(255)
STOP8_TIME : time
TRAVEL8_TIME : int(11)
STOP9_NAME : varchar(255)
STOP9_TIME : time
TRAVEL9_TIME : int(11)
STOP10_NAME : varchar(255)
STOP10_TIME : time
TRAVEL10_TIME : int(11)

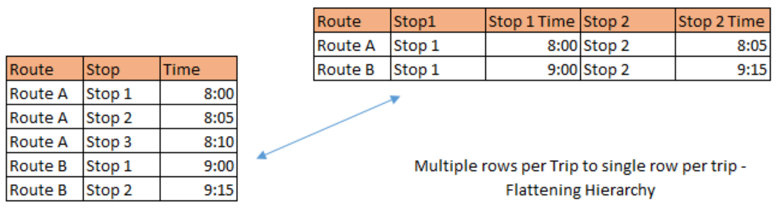
W_SHED_D
integration_key : varchar(35)
schedule_id : mediumtext
trip_id : int(11)
DOW : mediumtext
STOP1_NAME : varchar(255)
STOP1_TIME : time
STOP_1ID : int(11)
STOP2_NAME : varchar(255)
STOP2_TIME : time
STOP2_ID : int(11)
STOP3_NAME : varchar(255)
STOP3_TIME : time
STOP3_ID : int(11)
STOP4_NAME : varchar(255)
STOP4_TIME : time
STOP4_ID : int(11)
STOP5_NAME : varchar(255)
STOP5_TIME : time
STOP5_ID : int(11)
STOP6_NAME : varchar(255)
STOP6_TIME : time
STOP6_ID : int(11)
STOP7_NAME : varchar(255)
STOP7_TIME : time
STOP7_ID : int(11)
STOP8_NAME : varchar(255)
STOP8_TIME : time
STOP8_ID : int(11)
STOP9_NAME : varchar(255)
STOP9_TIME : time
STOP9_ID : int(11)
STOP10_NAME : varchar(255)
STOP10_TIME : time
STOP10_ID : int(11)

F_TRIPS_COMPARED
SCHEDULE_ID : mediumtext
TRIP_ID : int(11)
Date : date
Inbound : int(11)
outbound : int(11)
schedule_start : time
actual_start : time
schedule_end : time
actual_end : time

It can be seen that instead of a one-row per stop arroach, this data model now has all trip data in one row. The data model is designed in such a way that it can hold upto 30 stops and a select statement can be used to query the data from these tables and calculate variance.

4.4 Data Preparation

As discussed above, the data is now available in OLTP tables with the trip id and stop order. Now asper the data model the data has to be flattened to the new structure.



Route	Stop	Time
Route A	Stop 1	8:00
Route A	Stop 2	8:05
Route A	Stop 3	8:10
Route B	Stop 1	9:00
Route B	Stop 2	9:15

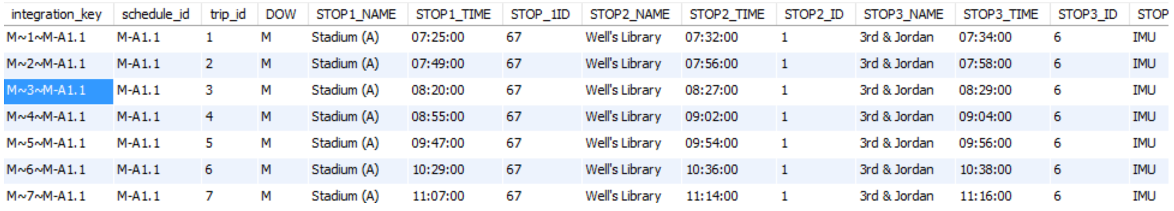
Route	Stop1	Stop 1 Time	Stop 2	Stop 2 Time
Route A	Stop 1	8:00	Stop 2	8:05
Route B	Stop 1	9:00	Stop 2	9:15

Multiple rows per Trip to single row per trip - Flattening Hierarchy

As shown in the above the transformation is made by repeated joins to the same table. For the schedule table,

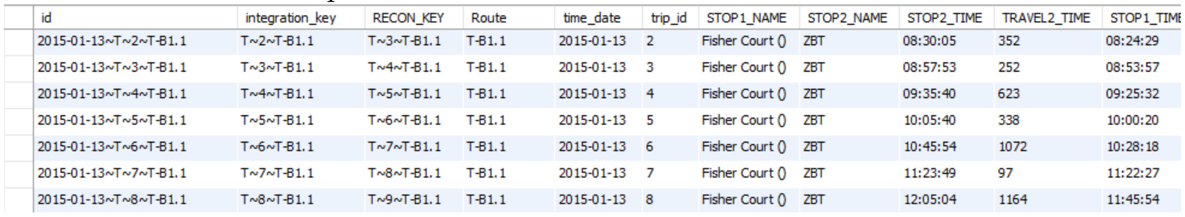
```
concat(DOW,concat('~',concat(trip_id,concat('~',Schedule_ID)))) 'integration_key'
```

is used as an integration key. This is nothing but the concatenation of day of week, trip id, and the route if concatenated to create an unique key for every trip. Thus the data is available on the schedule dimension as below,



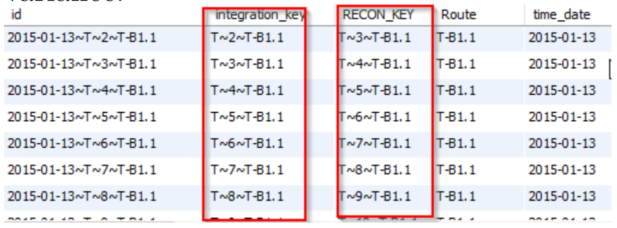
integration_key	schedule_id	trip_id	DOW	STOP1_NAME	STOP1_TIME	STOP_1ID	STOP2_NAME	STOP2_TIME	STOP2_ID	STOP3_NAME	STOP3_TIME	STOP3_ID	STOP
M~1~M-A1.1	M-A1.1	1	M	Stadium (A)	07:25:00	67	Well's Library	07:32:00	1	3rd & Jordan	07:34:00	6	IMU
M~2~M-A1.1	M-A1.1	2	M	Stadium (A)	07:49:00	67	Well's Library	07:56:00	1	3rd & Jordan	07:58:00	6	IMU
M~3~M-A1.1	M-A1.1	3	M	Stadium (A)	08:20:00	67	Well's Library	08:27:00	1	3rd & Jordan	08:29:00	6	IMU
M~4~M-A1.1	M-A1.1	4	M	Stadium (A)	08:55:00	67	Well's Library	09:02:00	1	3rd & Jordan	09:04:00	6	IMU
M~5~M-A1.1	M-A1.1	5	M	Stadium (A)	09:47:00	67	Well's Library	09:54:00	1	3rd & Jordan	09:56:00	6	IMU
M~6~M-A1.1	M-A1.1	6	M	Stadium (A)	10:29:00	67	Well's Library	10:36:00	1	3rd & Jordan	10:38:00	6	IMU
M~7~M-A1.1	M-A1.1	7	M	Stadium (A)	11:07:00	67	Well's Library	11:14:00	1	3rd & Jordan	11:16:00	6	IMU

Once the data to schedule is loaded, now the data to the actual run can be loaded. However to this new table, the unique key is the concatenation of data and the other components as described above.



id	integration_key	RECON_KEY	Route	time_date	trip_id	STOP1_NAME	STOP2_NAME	STOP2_TIME	TRAVEL2_TIME	STOP1_TIME
2015-01-13~T~2~T-B1.1	T~2~T-B1.1	T~3~T-B1.1	T-B1.1	2015-01-13	2	Fisher Court ()	ZBT	08:30:05	352	08:24:29
2015-01-13~T~3~T-B1.1	T~3~T-B1.1	T~4~T-B1.1	T-B1.1	2015-01-13	3	Fisher Court ()	ZBT	08:57:53	252	08:53:57
2015-01-13~T~4~T-B1.1	T~4~T-B1.1	T~5~T-B1.1	T-B1.1	2015-01-13	4	Fisher Court ()	ZBT	09:35:40	623	09:25:32
2015-01-13~T~5~T-B1.1	T~5~T-B1.1	T~6~T-B1.1	T-B1.1	2015-01-13	5	Fisher Court ()	ZBT	10:05:40	338	10:00:20
2015-01-13~T~6~T-B1.1	T~6~T-B1.1	T~7~T-B1.1	T-B1.1	2015-01-13	6	Fisher Court ()	ZBT	10:45:54	1072	10:28:18
2015-01-13~T~7~T-B1.1	T~7~T-B1.1	T~8~T-B1.1	T-B1.1	2015-01-13	7	Fisher Court ()	ZBT	11:23:49	97	11:22:27
2015-01-13~T~8~T-B1.1	T~8~T-B1.1	T~9~T-B1.1	T-B1.1	2015-01-13	8	Fisher Court ()	ZBT	12:05:04	1164	11:45:54

However, even with this, we would not be able to match the schedule and run. It might happen that for a given trip, the bus may not have stopped in stop 1, hence the whole trip would be lost. Hence we calculate an reconciliation key, $RECON_KEY$. It can be seen that these two keys need not be same and the new key can be used to join with the schedule table to calculate variance.

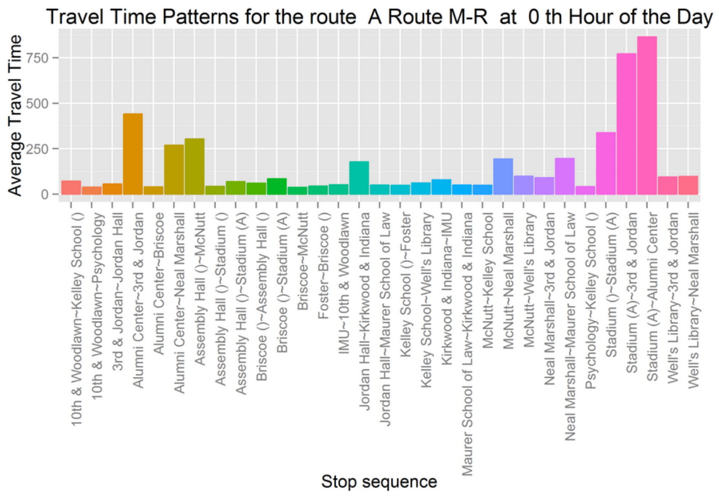


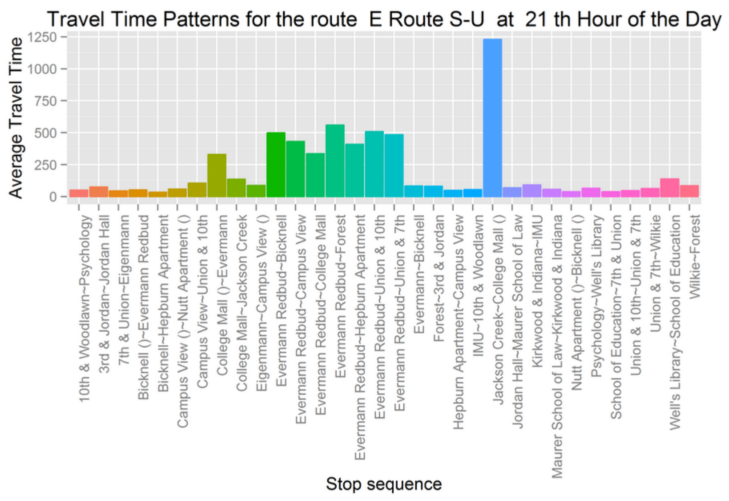
id	integration_key	RECON_KEY	Route	time_date
2015-01-13~T~2~T-B1.1	T~2~T-B1.1	T~3~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~3~T-B1.1	T~3~T-B1.1	T~4~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~4~T-B1.1	T~4~T-B1.1	T~5~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~5~T-B1.1	T~5~T-B1.1	T~6~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~6~T-B1.1	T~6~T-B1.1	T~7~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~7~T-B1.1	T~7~T-B1.1	T~8~T-B1.1	T-B1.1	2015-01-13
2015-01-13~T~8~T-B1.1	T~8~T-B1.1	T~9~T-B1.1	T-B1.1	2015-01-13

The variance was just then the select away and is calculated directly.

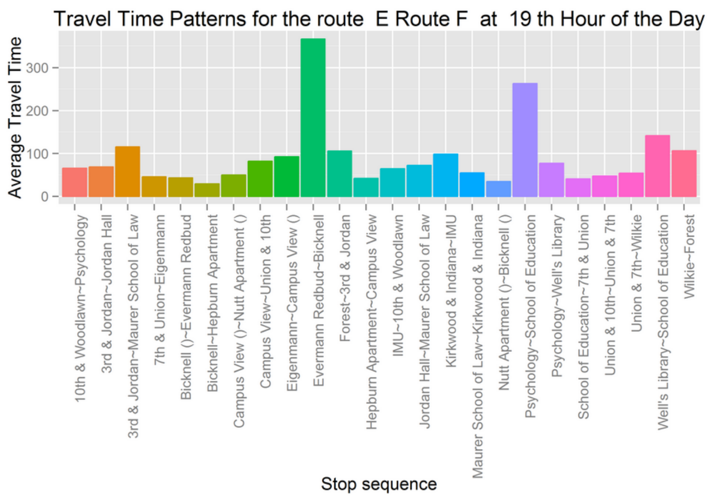
5 Exploratory Analysis

Any Start of Data Analytics involves understanding of the data. An initial Exploratory data analysis was made using R and ggplot2 library. For example the below graph indicates the average time between the pairs of stops for the 12 AM for Route A. It can be seen that the travelling time is very less during morning 12AM.

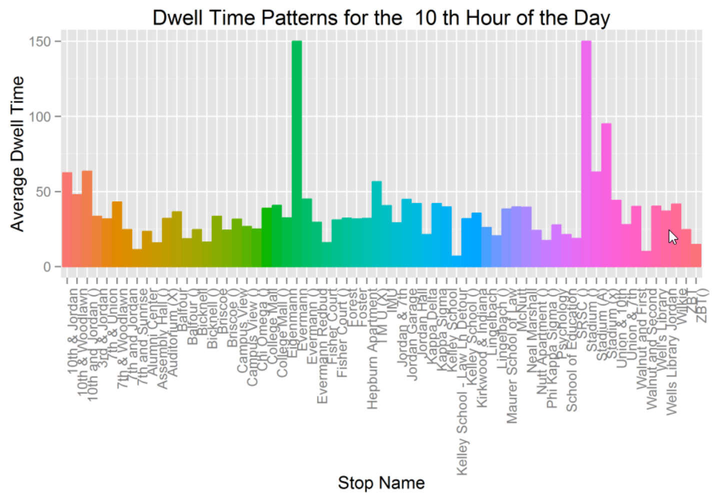




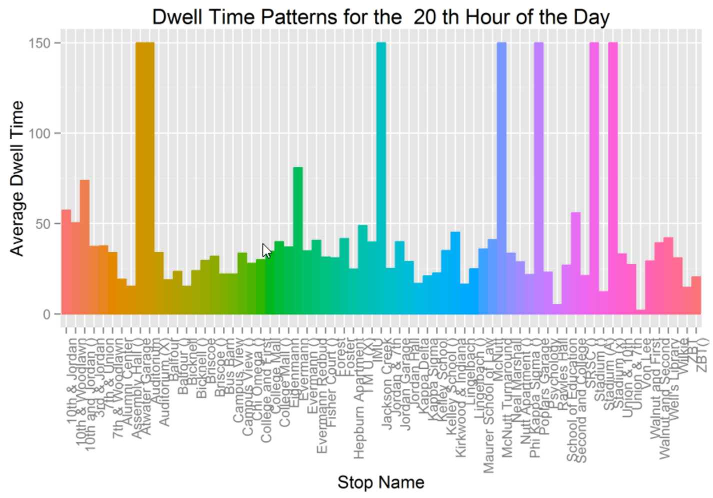
However, a change in trend happens during the ”peak times”. During evening 7, the average time traveling time increases drastically, for Eg, 10th and Wood Lawn to psychology takes less than a minute during morning, however during evenings, it double to about 2 minutes. The same trend can be seen at below plots.



However things get different when we talk about the dwell time. The average dwell time was plotted for every single hour of the day, it is interesting to note that the maximum time the bus spends is in the originating stops!



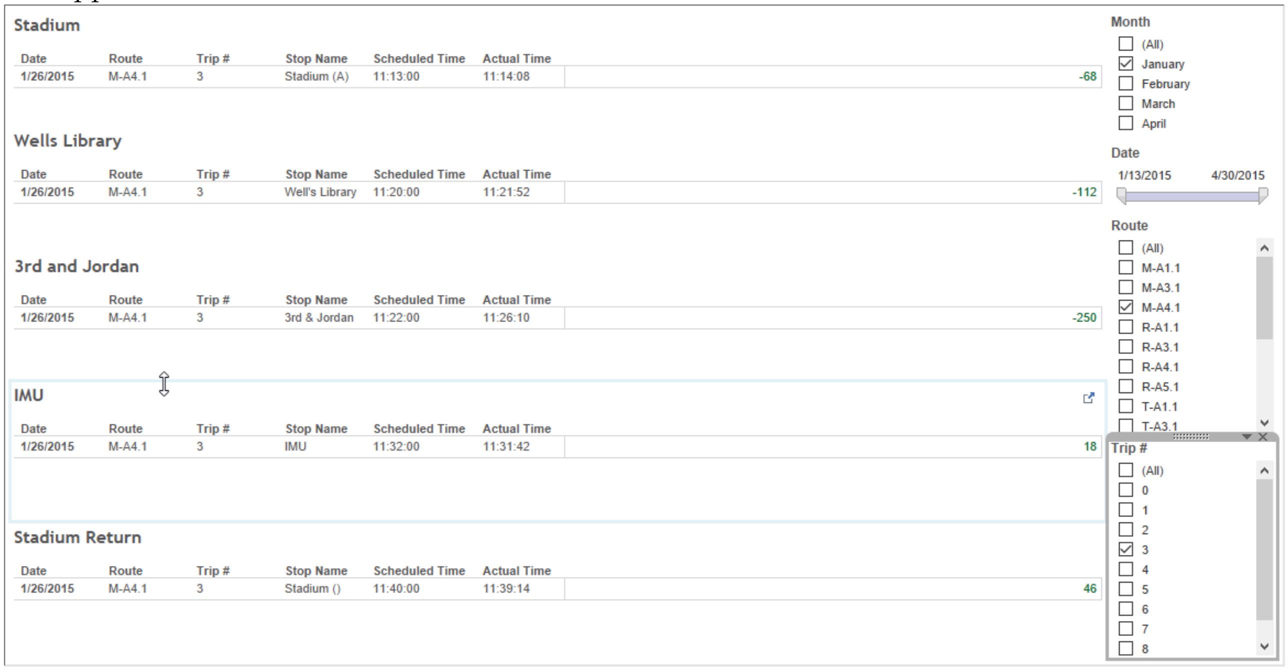
Another interesting trend is that as the day progresses, dwell time increases in major stops. There is in fact spike in the major stops like IMU, Kirkwood and Indiana and Kelley's school of Business.



All these graphs and more visualizations are available in the Github Repository, `https://github.com/sarvothaman/nefarious-octo-rutabaga/tree/master/EDA`

6 Reporting

The Main objective of this case study is to report how much the busses run in accordance to the schedule. ie, the time deviation $t_{schedule} - t_{acutal}$ is an important metric. Tableau reporting solution was used to report these findings.
Tableau provides an excellent platform for developing these reports. The variance reports are reports which gives the flexibility to select the month, data, route and trip and check the variance in each stops. This is available on cloud and can be accessed by anyone for free. Below is the screenshot of the application hosted.



Stadium

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Stadium (A)	11:13:00	11:14:08	-68

Wells Library

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Well's Library	11:20:00	11:21:52	-112

3rd and Jordan

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	3rd & Jordan	11:22:00	11:26:10	-250

IMU

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	IMU	11:32:00	11:31:42	18

Stadium Return

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
1/26/2015	M-A4.1	3	Stadium ()	11:40:00	11:39:14	46

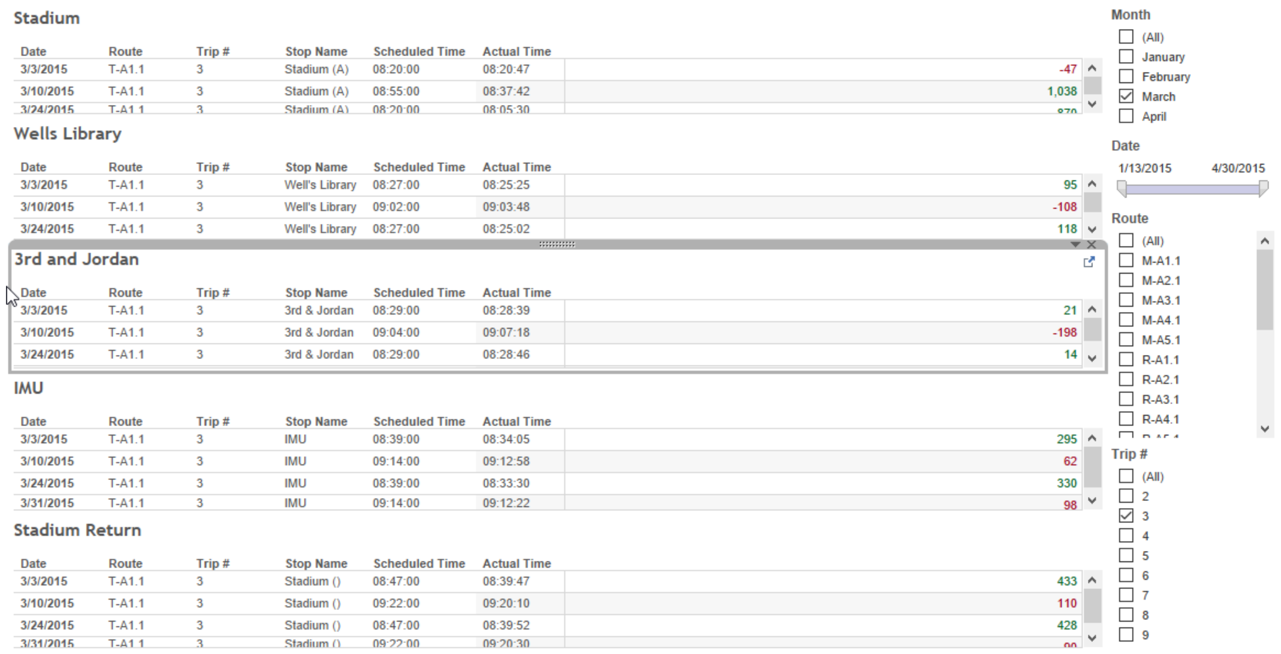
Month
- (All)
- [✓ January]
- February
- March
- April

Date
1/13/2015 4/30/2015

Route
- (All)
- M-A1.1
- M-A3.1
- [✓ M-A4.1]
- R-A1.1
- R-A3.1
- R-A4.1
- R-A5.1
- T-A1.1
- T-A3.1

Trip #
- (All)
- 0
- 1
- 2
- [✓ 3]
- 4
- 5
- 6
- 7
- 8

For example, the above shown trip is the 3 trip of A1 on January 26th 2015. It can be seen that the departure from stadium A is at 11:13, however the bus departed at 11:14 bringing in an one minute delay. It is interesting that in 3rd and Jordan, the was about 3 minutes late, however it reached IMU one minute early. This suggests that there is still space for optimizing the schedule. Multiple trips can also be selected as shown below to access trip information.



Stadium

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Stadium (A)	08:20:00	08:20:47	-47
3/10/2015	T-A1.1	3	Stadium (A)	08:55:00	08:37:42	1,038
3/24/2015	T-A1.1	3	Stadium (A)	08:20:00	08:05:30	870

Wells Library

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Well's Library	08:27:00	08:25:25	95
3/10/2015	T-A1.1	3	Well's Library	09:02:00	09:03:48	-108
3/24/2015	T-A1.1	3	Well's Library	08:27:00	08:25:02	118

3rd and Jordan

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	3rd & Jordan	08:29:00	08:28:39	21
3/10/2015	T-A1.1	3	3rd & Jordan	09:04:00	09:07:18	-198
3/24/2015	T-A1.1	3	3rd & Jordan	08:29:00	08:28:46	14

IMU

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	IMU	08:39:00	08:34:05	295
3/10/2015	T-A1.1	3	IMU	09:14:00	09:12:58	62
3/24/2015	T-A1.1	3	IMU	08:39:00	08:33:30	330
3/31/2015	T-A1.1	3	IMU	09:14:00	09:12:22	98

Stadium Return

Date	Route	Trip #	Stop Name	Scheduled Time	Actual Time	
3/3/2015	T-A1.1	3	Stadium ()	08:47:00	08:39:47	433
3/10/2015	T-A1.1	3	Stadium ()	09:22:00	09:20:10	110
3/24/2015	T-A1.1	3	Stadium ()	08:47:00	08:39:52	428
3/31/2015	T-A1.1	3	Stadium ()	09:22:00	09:20:30	90

Month
- (All)
- January
- February
- [✓ March]
- April

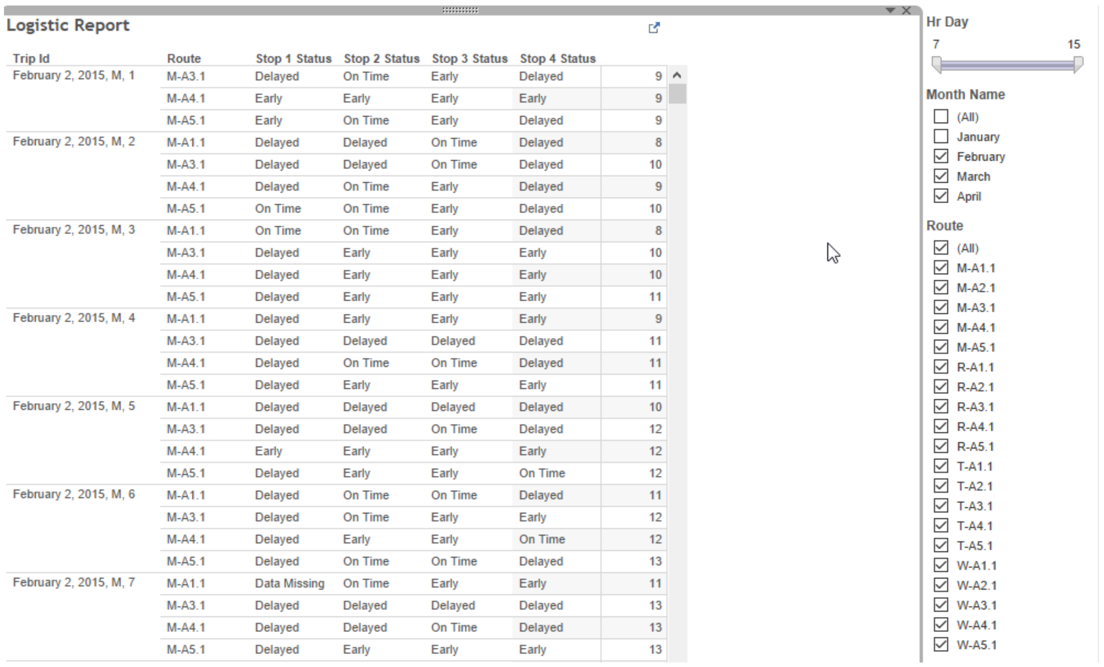
Date
1/13/2015 — 4/30/2015

Route
- (All)
- M-A1.1
- M-A2.1
- M-A3.1
- M-A4.1
- M-A5.1
- R-A1.1
- R-A2.1
- R-A3.1
- R-A4.1

Trip #
- (All)
- 2
- [✓ 3]
- 4
- 5
- 6
- 7
- 8
- 9

Status Reports

Sometimes we are interested in seeing that only whether the bus has arrived on-time or early to the main stops. The below report is called the logistic report and is available on cloud. This gives the user the option to drill down a specific date, month, route then trip and displays the status.



Logistic Report

Trip Id	Route	Stop 1 Status	Stop 2 Status	Stop 3 Status	Stop 4 Status	
February 2, 2015, M, 1	M-A3.1	Delayed	On Time	Early	Delayed	9
	M-A4.1	Early	Early	Early	Early	9
	M-A5.1	Early	On Time	Early	Delayed	9
February 2, 2015, M, 2	M-A1.1	Delayed	Delayed	On Time	Delayed	8
	M-A3.1	Delayed	Delayed	On Time	Delayed	10
	M-A4.1	Delayed	On Time	Early	Delayed	9
	M-A5.1	On Time	On Time	Early	Delayed	10
February 2, 2015, M, 3	M-A1.1	On Time	On Time	Early	Delayed	8
	M-A3.1	Delayed	Early	Early	Early	10
	M-A4.1	Delayed	Early	Early	Early	10
	M-A5.1	Delayed	Early	Early	Early	11
February 2, 2015, M, 4	M-A1.1	Delayed	Early	Early	Early	9
	M-A3.1	Delayed	Delayed	Delayed	Delayed	11
	M-A4.1	Delayed	On Time	On Time	Delayed	11
	M-A5.1	Delayed	Early	Early	Early	11
February 2, 2015, M, 5	M-A1.1	Delayed	Delayed	Delayed	Delayed	10
	M-A3.1	Delayed	Delayed	On Time	Delayed	12
	M-A4.1	Early	Early	Early	Early	12
	M-A5.1	Delayed	Early	Early	On Time	12
February 2, 2015, M, 6	M-A1.1	Delayed	On Time	On Time	Delayed	11
	M-A3.1	Delayed	On Time	Early	Early	12
	M-A4.1	Delayed	Early	Early	On Time	12
	M-A5.1	Delayed	On Time	On Time	Delayed	13
February 2, 2015, M, 7	M-A1.1	Data Missing	On Time	Early	Early	11
	M-A3.1	Delayed	Delayed	Delayed	Delayed	13
	M-A4.1	Delayed	Delayed	On Time	Delayed	13
	M-A5.1	Delayed	Early	Early	Early	13

Hr Day
7 15

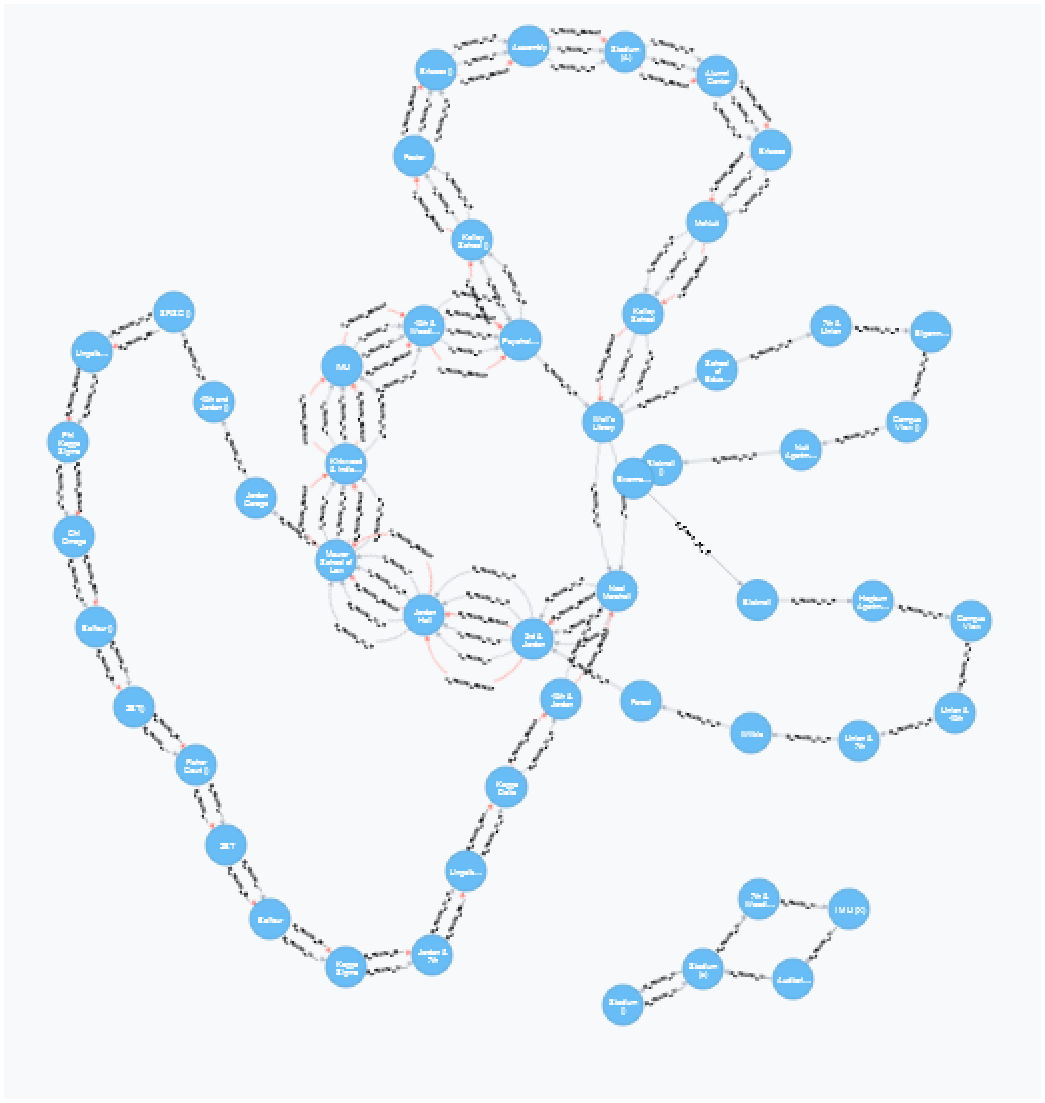
Month Name
☐ (All)
☐ January
[✓ February]
[✓ March]
[✓ April]

Route
[✓ (All)]
[✓ M-A1.1]
[✓ M-A2.1]
[✓ M-A3.1]
[✓ M-A4.1]
[✓ M-A5.1]
[✓ R-A1.1]
[✓ R-A2.1]
[✓ R-A3.1]
[✓ R-A4.1]
[✓ R-A5.1]
[✓ T-A1.1]
[✓ T-A2.1]
[✓ T-A3.1]
[✓ T-A4.1]
[✓ T-A5.1]
[✓ W-A1.1]
[✓ W-A2.1]
[✓ W-A3.1]
[✓ W-A4.1]
[✓ W-A5.1]

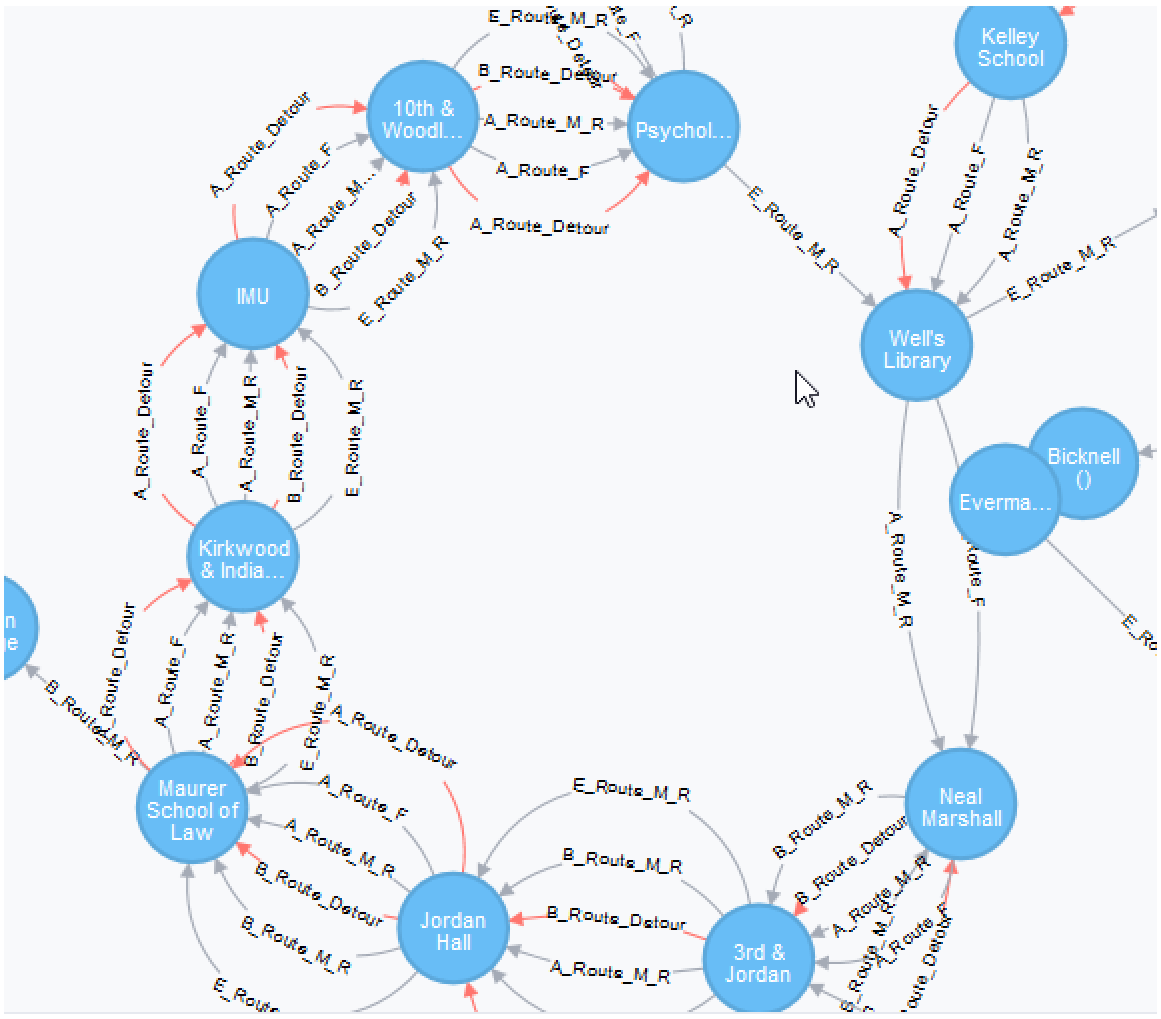
For example the Feburary 2nd, A4 route, 1st trip reached all its stop ahead time.
Thus care has been taken during this case study that due importance is given to usability of the output artifacts and can be used by the business rather than the esoteric data miners.

7 Visualization

Another aspect of the data is the ability to visualize the data to understand the route. Neo4j is used as the background data store for saving node data. pyMysql module was used to connect to mysql and py2neo was used to connect to the cloud instance.
A python program was written to aggregate the data from MySql and load it to cloud Neo4j instance. Once the entire data is loaded, the graph looks as follows.



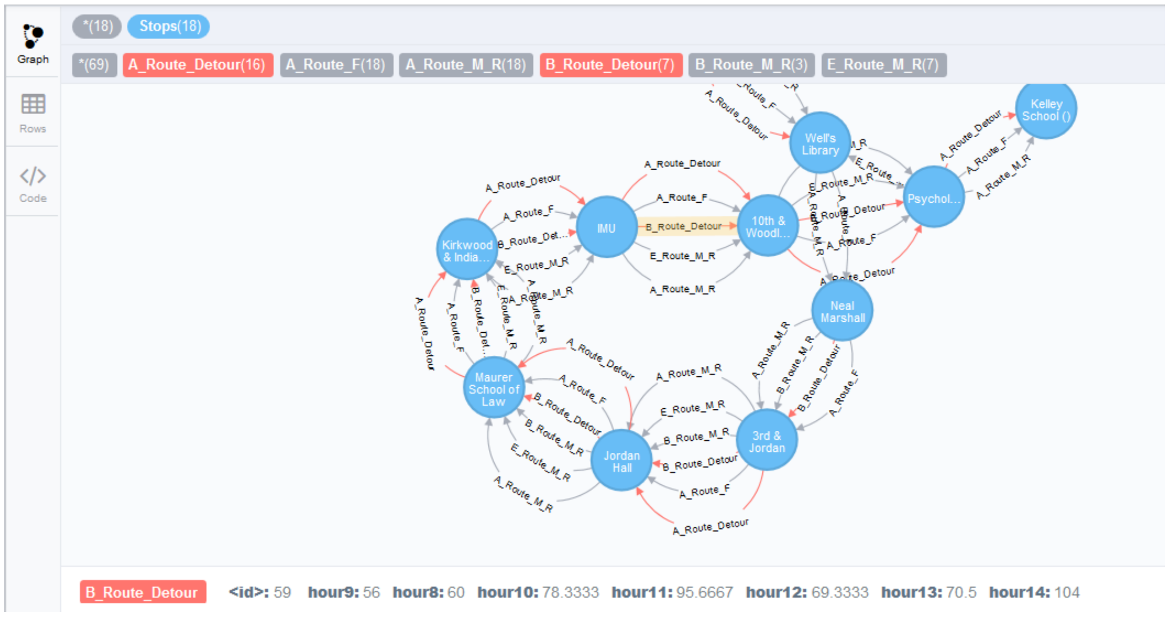
A close in screenshot is as follows. The data model is created in such a way that each stops are represented as nodes and each directed arrow is the route. These edges has a properties like travel time at any given hour of the day and the nodes contain average dwell time in that stop for a given day



Neo4j uses the language called Cypher instead of the SQL query. An example cypher query is as follows

```
MATCH p =(a:Stops { name: "Neal Marshall" })
-[:A_Route_M_R *]-
(b:Stops { name: "IMU" })
RETURN p, length(p)
```

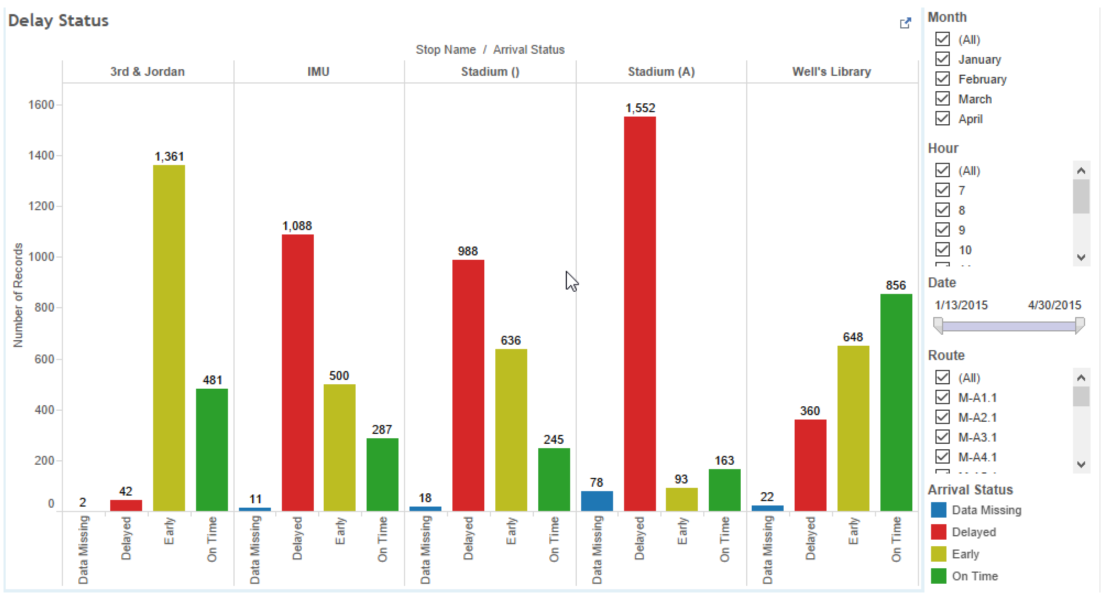
The above cypher selects a path from Neal Marshall to IMU and follows the route A in the Monday to Thursday schedule.



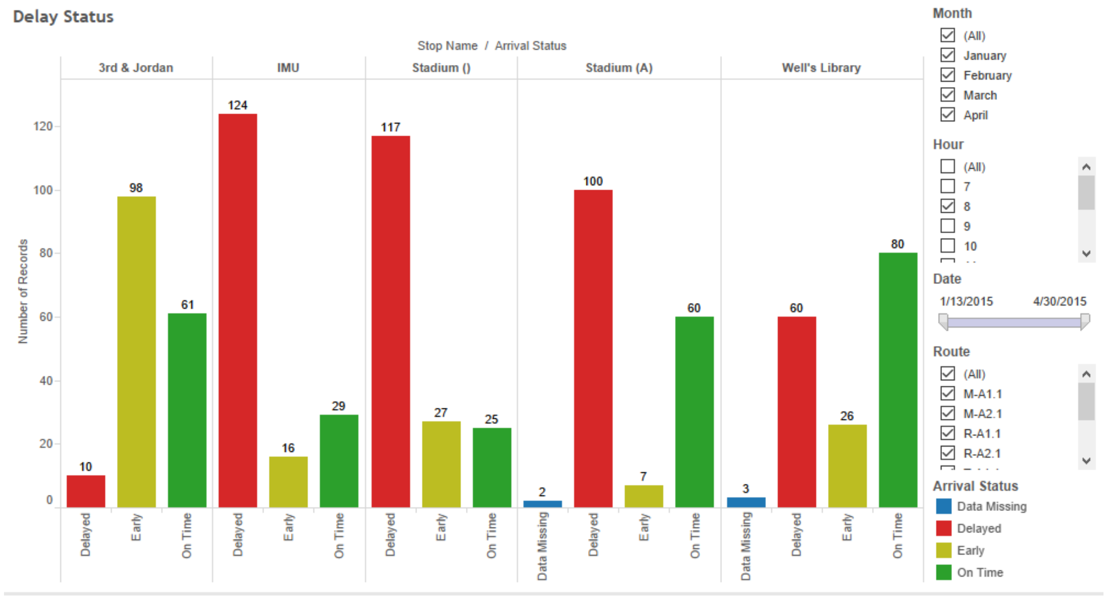
B_Route_Detour **<id>:** 59 **hour9:** 56 **hour8:** 60 **hour10:** 78.3333 **hour11:** 95.6667 **hour12:** 69.3333 **hour13:** 70.5 **hour14:** 104

It can be seen below that there are parameters like hour9 indicates the travelling time at 9th hour of the day. This was used to predict the delay/arrival time and most of the time it predicted the time accurate upto Plus or minus three minutes.

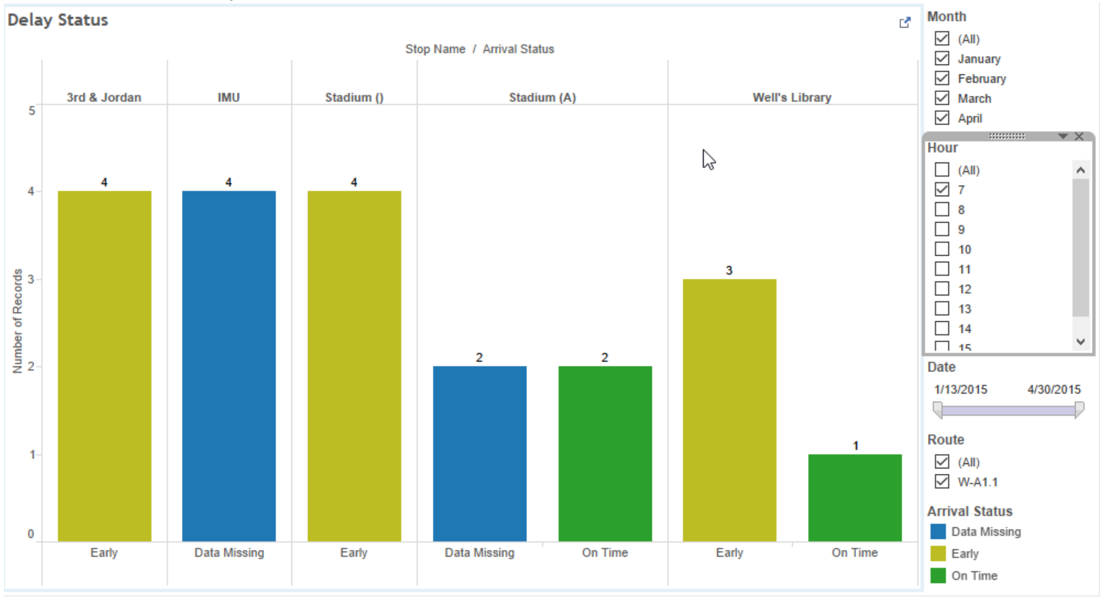
Status Reporting

As seen in above reports, for effective decision making the counts of instances bus statuses over a period of time is important. The below plots help in such interpretation, 

Eg, the above graph gives the count of the instances of the statuses. Generally it can be seen that stadium is always delayed and wells library is always on time.



This also lets the user to select the hour of the day, month, day, route and see how the delay status is affected.



In refreshing change, During the mornings, for most of the stops the bus reaches either early or on time.

8 Statistical Model

9 Future Work

10 Concluding remarks

11 References