

Leeds Beckett University  
Faculty of Arts, Environment & Technology

*MSc Business Intelligence*

*Academic Year 2015-2016*

*Joanne Kennedy C3369865*

*Data Warehouse Models and Approaches:*

*Implementation of PlaceU Data Warehouse System*

*Date of Submission: 8<sup>th</sup> May 2016*

# *Contents*

## **Article I.** Introduction

## **Article II.** Data Collection and Cleansing (ETL)

2.01 Original Datasets

2.02 Star Schema

2.03 ETL

## **Article III.** Data Analysis

## **Article IIV.** OLAP Using Excel

## **Article V.** Findings & Evaluation

## **Article VI.** Bibliography

## **Article VI.** Appendix

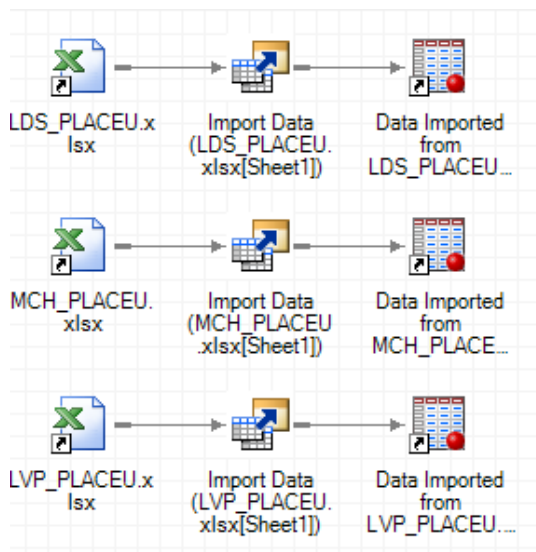
7.01 SAS Code

## Article I. Introduction

This report will evidence and explain the process for data warehouse implementation. The steps that have been undertaken supports the data warehouse requirements and reports which were created in part 1 (data warehouse design approach assignment) along with the incorporation of appropriate feedback.

## Article II. Data Collection and Cleansing (ETL)

### Section 2.01 Original Data Sets



Datasets were created in SAS with the use of the import wizard task which allows SAS to automatically generate code. Three sources were used which created three SAS datasets, one for consultants from the Leeds branch, one for consultants from the Manchester branch and one for consultants from the Liverpool branch. The code for the imported data was then slightly modified to ensure the data would be stored in the PlaceU library as a pose to the temporary work library.

The three tables with already a sufficient amount of populated data can be seen running successfully below.

	CONSULTANT_ID	CST_NAME	CST_START	CST_END	LOCATION_ID	LOCATION_NAME	LOCATION_POST
1	1	Aaron Abbotts	22JUN2003	.	1	Leeds	L1 4HR
2	2	Ben Bunning	22MAY2014	.	1	Leeds	L1 4HR
3	3	Charlie Crumble	22JUN2009	.	1	Leeds	L1 4HR
4	4	Dan Dare	16FEB2000	.	1	Leeds	L1 4HR
5	5	Elliot Evans	14JUN2012	.	1	Leeds	L1 4HR
6	6	Fred Frump	22JUN2010	.	1	Leeds	L1 4HR
7	7	Gilly Green	07OCT2011	.	1	Leeds	L1 4HR
8	8	Harry Hoo	22JUN2012	.	1	Leeds	L1 4HR
9	9	Izzac Ingle	22JAN2009	.	1	Leeds	L1 4HR
10	10	Jenna Jenkin	22JUN2000	.	1	Leeds	L1 4HR
11	11	Ken Kettle	20APR2015	.	1	Leeds	L1 4HR

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Process Flow

Project Tree

- LDS\_PLACEU.xlsx
  - Import Data (LDS\_PLACEU.xlsx[Sheet1])
- MCH\_PLACEU.xlsx
  - Import Data (MCH\_PLACEU.xlsx[Sheet1])
- LVP\_PLACEU.xlsx
  - Import Data (LVP\_PLACEU.xlsx[Sheet1])
- Programs
  - Program
  - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

Refresh Disconnect Stop

Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])

Filter and Sort Query Builder Where Data Describe Graph Analyze Export Send To

	CONSULTANT_ID	CST_FNAME	CST_SNAME	CST_START	CST_END	LOCATION_ID	LOCATION_NAME	LOCATION_POST
1	1	Jake	Abbot	22JUN2003	.	2	Manchester	M11 3FF
2	2	Boris	Burnings	22MAY2014	.	2	Manchester	M11 3FF
3	3	Cleo	Crumbled	22JUN2009	.	2	Manchester	M11 3FF
4	4	Danny	Dared	16FEB2000	22JUL2005	2	Manchester	M11 3FF
5	5	Ellie	Evan	14JUN2012	.	2	Manchester	M11 3FF
6	6	Freddie	Frumped	22JUN2010	.	2	Manchester	M11 3FF
7	7	Gillian	Greenock	07OCT2011	.	2	Manchester	M11 3FF
8	8	Hamison	Hoot	22JUN2012	.	2	Manchester	M11 3FF
9	9	Imogen	Ingleby	22JAN2009	.	2	Manchester	M11 3FF
10	10	Jean	Joke	22JUN2000	22JUN2003	2	Manchester	M11 3FF
11	11	Kez	Kit	20APR2015	.	2	Manchester	M11 3FF

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Process Flow

Project Tree

- LDS\_PLACEU.xlsx
  - Import Data (LDS\_PLACEU.xlsx[Sheet1])
- MCH\_PLACEU.xlsx
  - Import Data (MCH\_PLACEU.xlsx[Sheet1])
- LVP\_PLACEU.xlsx
  - Import Data (LVP\_PLACEU.xlsx[Sheet1])
- Programs
  - Program
  - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

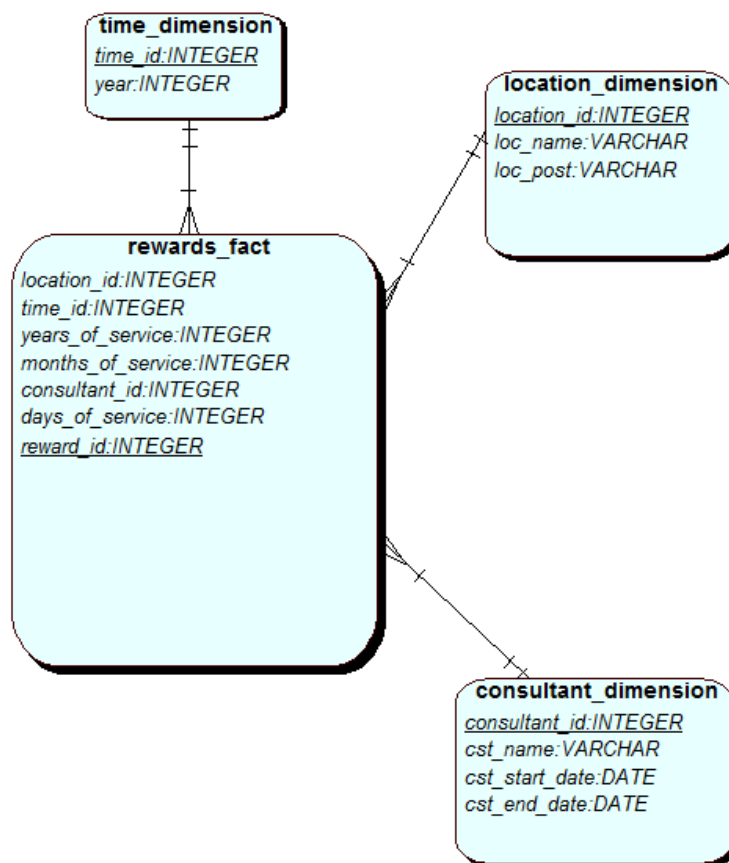
Refresh Disconnect Stop

Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

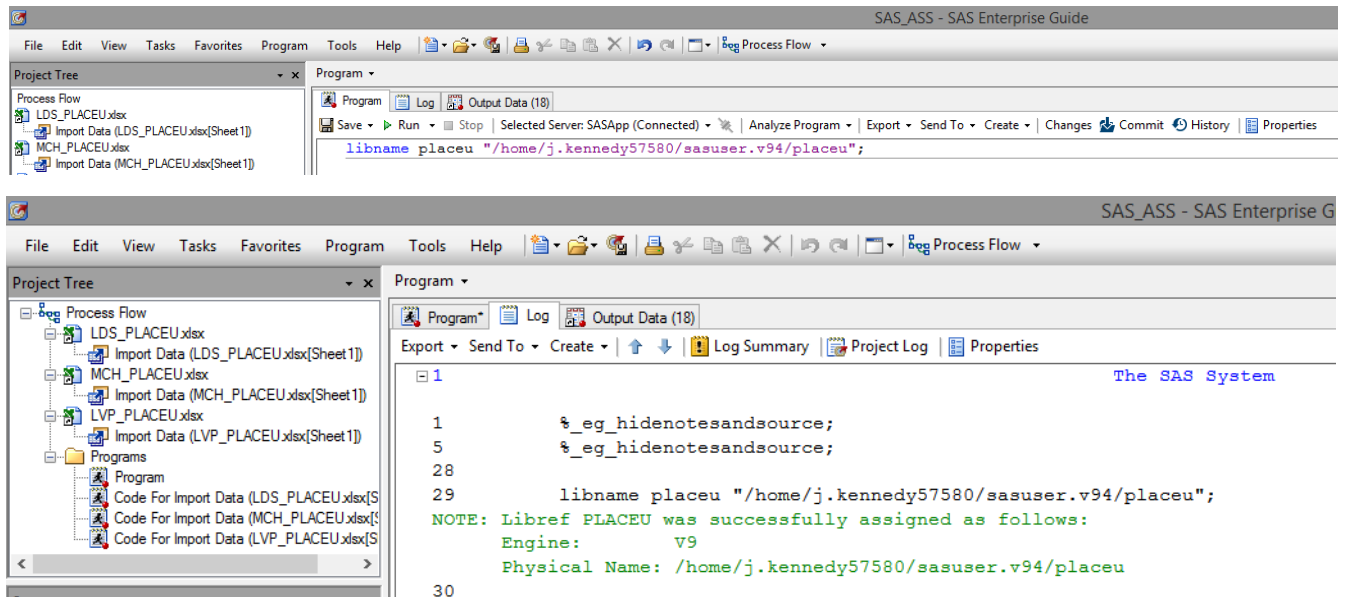
Filter and Sort Query Builder Where Data Describe Graph Analyze Export Send To

	CONSULTANT_ID	CST_FNAME	CST_SNAME	CST_START	CST_END	LOCATION_ID	LOCATION_NAME	LOCATION_POST
1	1	Abbie	Abet	22JUN2003	.	3	Liverpool	L1 4HR
2	2	Bordie	Brot	22MAY2014	.	3	Liverpool	L1 4HR
3	3	Champ	Citm	22JUN2009	.	3	Liverpool	L1 4HR
4	4	Drew	Drake	16FEB2000	.	3	Liverpool	L1 4HR
5	5	Enid	Eves	14JUN2012	.	3	Liverpool	L1 4HR
6	6	Freya	Fell	22JUN2010	.	3	Liverpool	L1 4HR
7	7	Gail	Grend	07OCT2011	.	3	Liverpool	L1 4HR
8	8	Hailey	Hoops	22JUN2012	.	3	Liverpool	L1 4HR
9	9	Lilly	Lolly	22JAN2009	.	3	Liverpool	L1 4HR
10	10	Julia	Juke	22JUN2000	.	3	Liverpool	L1 4HR
11	11	Keith	Kit	20APR2015	.	3	Liverpool	L1 4HR

## Section 2.02 Star Schema

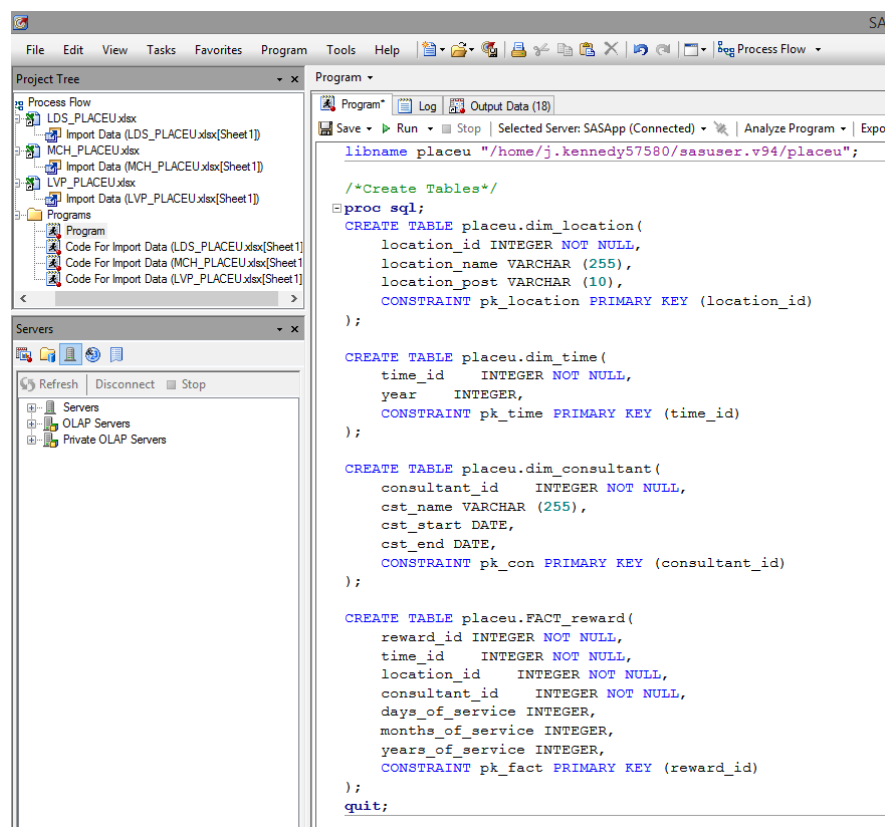


Above is the star scheme that will be implemented, there will be three dimension tables, a time dimension, a location dimension and a consultant dimension, along with a rewards fact table which will have its own unique key as a pose to the original star scheme design which used a compound key made up of the primary keys from each of the three dimensions.



To begin the implementation process, it was important to define the project library that would be used, so in the above example you can see the library placeu has been defined.

The next stage of the process was to create the empty dimension tables and fact table using the proc sql command which can be seen in the screenshot below.



The screenshot shows the SAS Studio interface with the following components:

- Project Tree:** Displays a process flow with steps for importing data from Excel files (LDS\_PLACEU.xlsx, MCH\_PLACEU.xlsx, LVP\_PLACEU.xlsx) and a program step.
- Servers:** Shows a list of servers including OLAP Servers and Private OLAP Servers.
- Program Editor:** Contains the following SAS code:
 

```

31 /*Create Tables*/
32 proc sql;
33     CREATE TABLE placeu.dim_location(
34         location_id INTEGER NOT NULL,
35         location_name VARCHAR (255),
36         location_post VARCHAR (10),
37         CONSTRAINT pk_location PRIMARY KEY (location_id)
38     );
39
40     CREATE TABLE placeu.dim_time(
41         time_id     INTEGER NOT NULL,
42         year        INTEGER,
43         CONSTRAINT pk_time PRIMARY KEY (time_id)
44     );
45
46     CREATE TABLE placeu.dim_consultant(
47         consultant_id INTEGER NOT NULL,
48         cst_name VARCHAR (255),
49         cst_start DATE,
50         cst_end DATE,
51         CONSTRAINT pk_con PRIMARY KEY (consultant_id)
52     );
53
54     CREATE TABLE placeu.FACT_reward(
55         reward_id INTEGER NOT NULL,
56         time_id     INTEGER NOT NULL,
57         location_id INTEGER NOT NULL,
58         consultant_id INTEGER NOT NULL,
59         days_of_service INTEGER,
60         months_of_service INTEGER,
61         years_of_service INTEGER,
62         CONSTRAINT pk_fact PRIMARY KEY (reward_id)
63     );
64 quit;

```
- Output Data (18):** Displays the execution results, including table creation messages and system statistics.
 

```

NOTE: Primary key pk_location replaced not null constraint _NM0001_.
NOTE: Table PLACEU.DIM_LOCATION created, with 0 rows and 3 columns.
NOTE: Primary key pk_time replaced not null constraint _NM0001_.
NOTE: Table PLACEU.DIM_TIME created, with 0 rows and 2 columns.
NOTE: Primary key pk_con replaced not null constraint _NM0001_.
NOTE: Table PLACEU.DIM_CONSULTANT created, with 0 rows and 4 columns.
NOTE: Primary key pk_fact replaced not null constraint _NM0001_.
NOTE: Table PLACEU.FACT_REWARD created, with 0 rows and 7 columns.
NOTE: PROCEDURE SQL used (Total process time):
      real time          0.10 seconds
      user cpu time      0.01 seconds
      system cpu time    0.01 seconds
      memory             623.28k
      OS Memory          20396.00k
      Timestamp          22/04/2016 03:28:22 PM
      Step Count         108   Switch Count  164

```

## Section 2.03 ETL

The ETL (Extract, Transform, Load) process is probably the most important stage in the development of a data warehouse.

The decision was made to manually insert data into the time dimension, however a sequence could have been used here for future additions, the code, output and log are as follows.

SAS Studio interface showing the Project Tree on the left, the Program window in the center, and the Servers window at the bottom. The Program window displays the following SQL code:

```
/*Manual Insert Into Time Dimension*/
proc sql;
  INSERT INTO placeu.dim_time
  VALUES (1, 2000);
  INSERT INTO placeu.dim_time
  VALUES (2, 2001);
  INSERT INTO placeu.dim_time
  VALUES (3, 2002);
  INSERT INTO placeu.dim_time
  VALUES (4, 2003);
  INSERT INTO placeu.dim_time
  VALUES (5, 2004);
  INSERT INTO placeu.dim_time
  VALUES (6, 2005);
  INSERT INTO placeu.dim_time
  VALUES (7, 2006);
  INSERT INTO placeu.dim_time
  VALUES (8, 2007);
  INSERT INTO placeu.dim_time
  VALUES (9, 2008);
  INSERT INTO placeu.dim_time
  VALUES (10, 2009);
  INSERT INTO placeu.dim_time
  VALUES (11, 2010);
  INSERT INTO placeu.dim_time
  VALUES (12, 2011);
  INSERT INTO placeu.dim_time
  VALUES (13, 2012);
  INSERT INTO placeu.dim_time
  VALUES (14, 2013);
  INSERT INTO placeu.dim_time
  VALUES (15, 2014);
  INSERT INTO placeu.dim_time
  VALUES (16, 2015);
  INSERT INTO placeu.dim_time
  VALUES (17, 2016);
quit;
```

SAS Studio interface showing the Project Tree on the left, the Program window in the center, and the DIM\_TIME table view on the right. The DIM\_TIME table contains the following data:

	time_id	year
1	1	2000
2	2	2001
3	3	2002
4	4	2003
5	5	2004
6	6	2005
7	7	2006
8	8	2007
9	9	2008
10	10	2009
11	11	2010
12	12	2011
13	13	2012
14	14	2013
15	15	2014
16	16	2015
17	17	2016

SAS Studio interface showing the Project Tree on the left, the Program window in the center, and the Log window at the bottom. The Log window displays the following output:

```
66
67      /*Manual Insert Into Time Dimension*/
68      proc sql;
69      INSERT INTO placeu.dim_time
70      VALUES (1, 2000);
71      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
72
73      INSERT INTO placeu.dim_time
74      VALUES (2, 2001);
75      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
76
77      INSERT INTO placeu.dim_time
78      VALUES (3, 2002);
79      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
80
81      INSERT INTO placeu.dim_time
82      VALUES (4, 2003);
83      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
84
85      INSERT INTO placeu.dim_time
86      VALUES (5, 2004);
87      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
88
89      INSERT INTO placeu.dim_time
90      VALUES (6, 2005);
91      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
92
93      INSERT INTO placeu.dim_time
94      VALUES (7, 2006);
95      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
96
97      INSERT INTO placeu.dim_time
98      VALUES (8, 2007);
99      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
100
101      INSERT INTO placeu.dim_time
102      VALUES (9, 2008);
103      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
104
105      INSERT INTO placeu.dim_time
106      VALUES (10, 2009);
107      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
108
109      INSERT INTO placeu.dim_time
110      VALUES (11, 2010);
111      NOTE: 1 row was inserted into PLACEU.DIM_TIME.
```

Due to the consultant data coming from three different sources and having multiple data discrepancies it was important to use a staging area.

```
/*Consultant Staging Area*/
proc sql;
Create table placeu.stagearea as select consultant_id, cst_name, cst_start, cst_end from placeu.lds_placeu;
Create table placeu.stageareaMan as select consultant_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from placeu.mch_placeu;
create table placeu.stageareaLiv as select consultant_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from placeu.lvp_placeu;
quit;

/*Merge Cleaned Consultant Data*/
data placeu.all_consultants;
set placeu.stagearea placeu.stageareaMan placeu.stageareaLiv;
run;

/*Create Consultant Sequence & Remove Missing Values*/
data placeu.all_consultants;
Set placeu.all_consultants;
consultant_id = _N_;
if (cst_end = .) then cst_end = today();
run;

/*Insert Into Consultant Dim*/
proc sql;
insert into placeu.dim_consultant select consultant_id, cst_name, cst_start, cst_end from placeu.all_consultants;
quit;
```

The screenshot displays the SAS Enterprise Guide interface. The left pane shows the Project Tree with a process flow diagram. The main window shows the SAS Program editor with the code from the previous block. The right pane shows the Log window, which contains the following output:

```
Page Swaps 0
Voluntary Context Switches 1055
Involuntary Context Switches 114
Block Input Operations 688
Block Output Operations 6800
```

```
104
105 /*Consultant Staging Area*/
106 proc sql;
107 Create table placeu.stagearea as select consultant_id, cst_name, cst_start, cst_end from placeu.lds_placeu;
NOTE: Table PLACEU.STAGEAREA created, with 11 rows and 4 columns.

108 Create table placeu.stageareaMan as select consultant_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end
108 ! from placeu.mch_placeu;
NOTE: Table PLACEU.STAGEAREAMAN created, with 11 rows and 4 columns.

109 create table placeu.stageareaLiv as select consultant_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end
109 ! from placeu.lvp_placeu;
NOTE: Table PLACEU.STAGEAREALIV created, with 11 rows and 4 columns.

110 quit;
NOTE: PROCEDURE SQL used (Total process time):
real time 0.04 seconds
```

Below the log output, the system status is displayed:

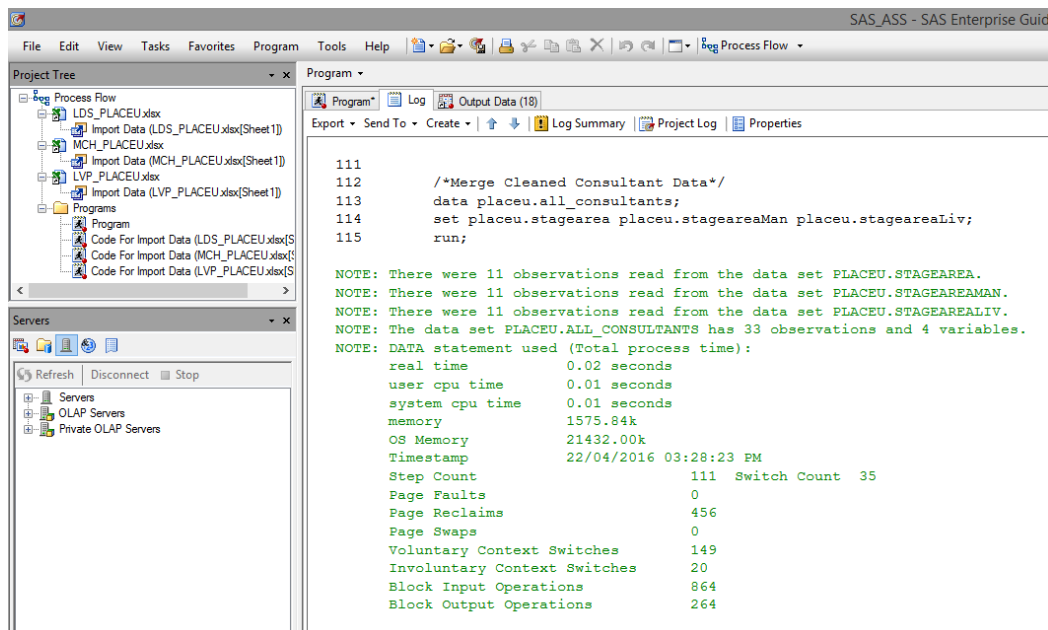
```
4 The SAS System 15:22 Friday, April 22, 2016
```

```
user cpu time 0.01 seconds
system cpu time 0.00 seconds
memory 5506.06k
OS Memory 25520.00k
Timestamp 22/04/2016 03:28:23 PM
Step Count 110 Switch Count 55
Page Faults 0
Page Reclaims 237
Page Swaps 0
Voluntary Context Switches 252
Involuntary Context Switches 16
Block Input Operations 0
Block Output Operations 792
```

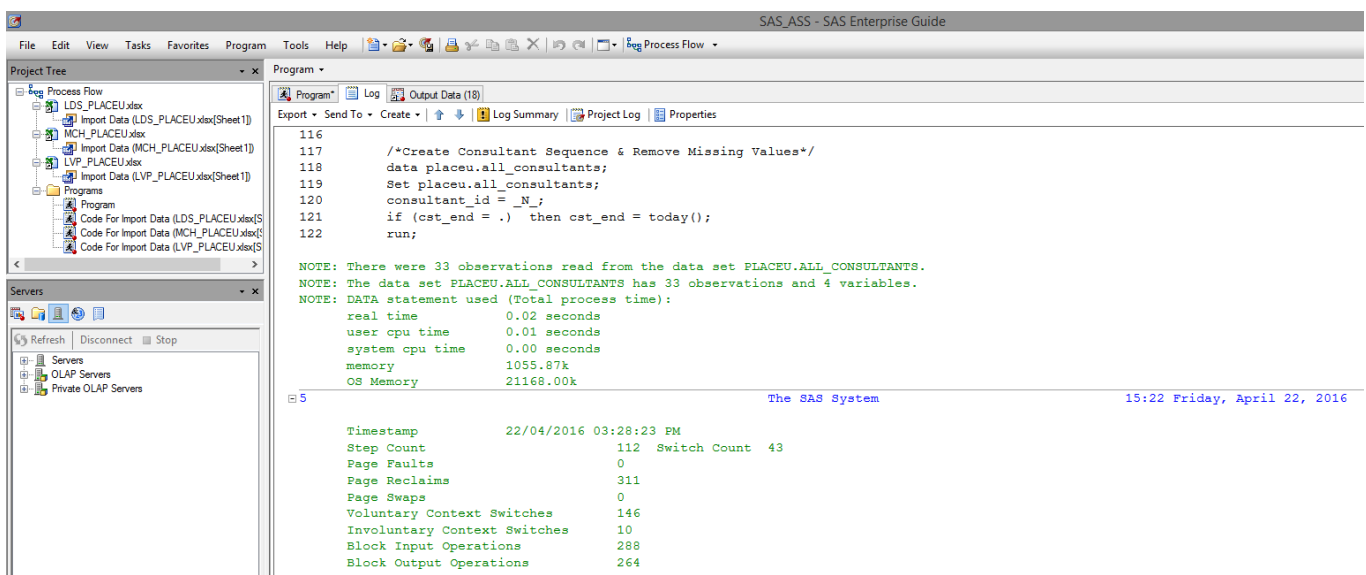
The above code creates three stage tables for each of the three data sources and uses a function to concatenate the consultants first names with their surnames from two of the sources due to the consultant dimension having one field for name.



A data step is then used to merge the cleaned data; however, it is important to note that at this stage we still have some missing values and the consultant id value is not yet unique.



Another data step is performed on the merged data in order to set up a sequence for the consultant id and ensure there are no missing values. The only missing values came from the consultant end date attribute so an 'if then' statement was used to ensure if the value was missing, today's date would be used as a substitute. This is using the logical assumption that if no end date was provided the consultant is still working for the company so as a tool for reporting, and to avoid any null values a 'today()' function was used.



After the extraction and transformation process was complete, the data could then be loaded into the consultant dimension, the output and log can be seen below.

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
    - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

Refresh Disconnect Stop

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Log Output Data (18)

DIM\_CONSULTANT

Filter and Sort Query Builder Where Data Describe Graph

	consultant_id	cst_name	cst_start	cst_end
1	1	Aaron Abbots	22JUN03	22APR16
2	2	Ben Bunning	22MAY14	22APR16
3	3	Charlie Crumble	22JUN09	22APR16
4	4	Dan Dare	16FEB00	22APR16
5	5	Elliot Evans	14JUN12	22APR16
6	6	Fred Frump	22JUN10	22APR16
7	7	Gilly Green	07OCT11	22APR16
8	8	Harry Hoo	22JUN12	22APR16
9	9	Izzac Ingle	22JAN09	22APR16
10	10	Jenna Jenkin	22JUN00	22APR16
11	11	Ken Kettle	20APR15	22APR16
12	12	Jake Abbot	22JUN03	22APR16
13	13	Boris Bunnings	22MAY14	22APR16
14	14	Cleo Crumbled	22JUN09	22APR16
15	15	Danny Dared	16FEB00	22JUL05
16	16	Ellie Evan	14JUN12	22APR16
17	17	Freddie Frumped	22JUN10	22APR16
18	18	Gillian Greenock	07OCT11	22APR16
19	19	Hamison Hoot	22JUN12	22APR16
20	20	Imogen Ingleby	22JAN09	22APR16
21	21	Jean Joke	22JUN00	22JUN03
22	22	Kez Kit	20APR15	22APR16
23	23	Abbie Abet	22JUN03	22APR16
24	24	Bordie Brot	22MAY14	22APR16
25	25	Champ Crim	22JUN09	22APR16
26	26	Drew Drake	16FEB00	22APR16
27	27	Enid Eves	14JUN12	22APR16
28	28	Freya Fell	22JUN10	22APR16
29	29	Gail Grend	07OCT11	22APR16
30	30	Hailey Hoops	22JUN12	22APR16
31	31	Lily Lolly	22JAN09	22APR16
32	32	Julia Juke	22JUN00	22APR16
33	33	Keith Kit	20APR15	22APR16

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
    - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

Refresh Disconnect Stop

Program

Log Output Data (18)

Export Send To Create Log Summary Project Log Properties

```

123
124      /*Insert Into Consultant Dim*/
125      proc sql;
126          insert into placeu.dim_consultant select consultant_id, cst_name, cst_start, cst_end from placeu.all_consultants;
NOTE: 33 rows were inserted into PLACEU.DIM_CONSULTANT.

127      quit;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.01 seconds
      user cpu time       0.00 seconds
      system cpu time     0.00 seconds
      memory              5948.68k
      OS Memory          26292.00k
      Timestamp           22/04/2016 03:28:23 PM
      Step Count          113  Switch Count  28
      Page Faults         0
      Page Reclaims       89
      Page Swaps          0
      Voluntary Context Switches 91
      Involuntary Context Switches 23
      Block Input Operations 976
      Block Output Operations 400
  
```

```
SAS_ASS - SAS Enterprise Guide

Tools Help | Save | Run | Stop | Selected Server: SASApp (Connected) | Analyze Program | Export | Send To | Create | Changes | Commit | History | Properties

Program* | Log | Output Data (18)

/*Location Staging Area*/
proc sql;
Create table placeu.placeu_location1 as select DISTINCT location_id, location_name, location_post from placeu.lds_placeu;
Create table placeu.placeu_location2 as select DISTINCT location_id, location_name, location_post from placeu.mch_placeu;
Create table placeu.placeu_location3 as select DISTINCT location_id, location_name, location_post from placeu.lvp_placeu;
quit;

/*Merge Cleaned Location Data*/
data placeu.placeu_locations;
set placeu.placeu_location1 placeu.placeu_location2 placeu.placeu_location3;
run;

/*Insert Into Location Dim*/
data placeu.dim_location ;
set placeu.placeu_locations ;
keep location_id location_name location_post;
run;
```

The next task was to do a similar thing for the location dimension due to the location information also being spread out over the three sources.

```
SAS_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help | Save | Run | Stop | Selected Server: SASApp (Connected) | Analyze Program | Export | Send To | Create | Changes | Commit | History | Properties

Project Tree | Program* | Log | Output Data (18) | Log Summary | Project Log | Properties

Process Flow
  LDS_PLACEU.xlsx
    Import Data (LDS_PLACEU.xlsx[Sheet1])
  MCH_PLACEU.xlsx
    Import Data (MCH_PLACEU.xlsx[Sheet1])
  LVP_PLACEU.xlsx
    Import Data (LVP_PLACEU.xlsx[Sheet1])
  Programs
    Program
      Code For Import Data (LDS_PLACEU.xlsx[Sheet1])
      Code For Import Data (MCH_PLACEU.xlsx[Sheet1])
      Code For Import Data (LVP_PLACEU.xlsx[Sheet1])

Servers
  Refresh Disconnect Stop
  Servers
  OLAP Servers
  Private OLAP Servers

128      /*Location Staging Area*/
129      proc sql;
130      Create table placeu.placeu_location1 as select DISTINCT location_id, location_name, location_post from placeu.lds_placeu;
NOTE: Table PLACEU.PLACEU_LOCATION1 created, with 1 rows and 3 columns.
131
132      Create table placeu.placeu_location2 as select DISTINCT location_id, location_name, location_post from placeu.mch_placeu;
NOTE: Table PLACEU.PLACEU_LOCATION2 created, with 1 rows and 3 columns.
133
134      Create table placeu.placeu_location3 as select DISTINCT location_id, location_name, location_post from placeu.lvp_placeu;
NOTE: Table PLACEU.PLACEU_LOCATION3 created, with 1 rows and 3 columns.
135
136      quit;
NOTE: PROCEDURE SQL used (Total process time):
      real time          0.05 seconds
      user cpu time      0.01 seconds
      system cpu time    0.00 seconds
      memory             5561.37k
      OS Memory          26032.00k
      Timestamp          22/04/2016 03:28:23 PM
      Step Count         114   Switch Count  47
      Page Faults        0
      Page Reclaims      224
      Page Swaps         0
      Voluntary Context Switches 233
      Involuntary Context Switches 15
      Block Input Operations 0
      Block Output Operations 792

6 The SAS System 15:22 Friday, April 22, 2016
```

Three stage tables was used for each of the three data sources to gather the location information, however due to multiple repeating values the ‘distinct’ statement was used to ensure only different values are returned.

```
SAS_ASS - SAS Enterprise Guide

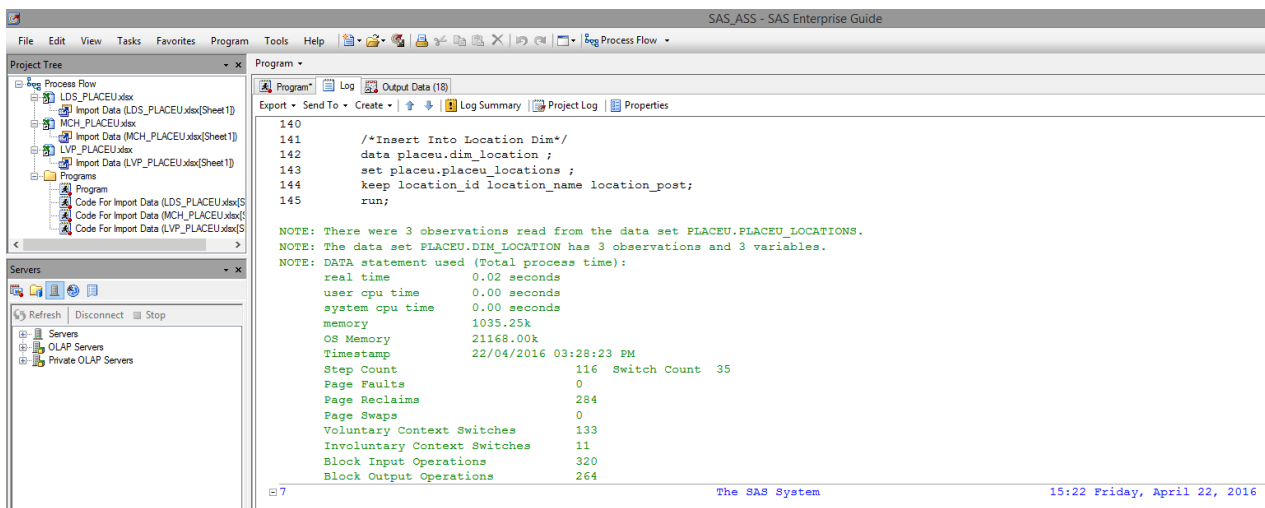
File Edit View Tasks Favorites Program Tools Help | Save | Run | Stop | Selected Server: SASApp (Connected) | Analyze Program | Export | Send To | Create | Changes | Commit | History | Properties

Project Tree | Program* | Log | Output Data (18) | Log Summary | Project Log | Properties

Process Flow
  LDS_PLACEU.xlsx
    Import Data (LDS_PLACEU.xlsx[Sheet1])
  MCH_PLACEU.xlsx
    Import Data (MCH_PLACEU.xlsx[Sheet1])
  LVP_PLACEU.xlsx
    Import Data (LVP_PLACEU.xlsx[Sheet1])
  Programs
    Program
      Code For Import Data (LDS_PLACEU.xlsx[Sheet1])
      Code For Import Data (MCH_PLACEU.xlsx[Sheet1])
      Code For Import Data (LVP_PLACEU.xlsx[Sheet1])

Servers
  Refresh Disconnect Stop
  Servers
  OLAP Servers
  Private OLAP Servers

135      /*Merge Cleaned Location Data*/
136      data placeu.placeu_locations;
137      set placeu.placeu_location1 placeu.placeu_location2 placeu.placeu_location3;
138      run;
139
NOTE: There were 1 observations read from the data set PLACEU.PLACEU_LOCATION1.
NOTE: There were 1 observations read from the data set PLACEU.PLACEU_LOCATION2.
NOTE: There were 1 observations read from the data set PLACEU.PLACEU_LOCATION3.
NOTE: The data set PLACEU.PLACEU_LOCATIONS has 3 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time          0.02 seconds
      user cpu time      0.01 seconds
      system cpu time    0.00 seconds
      memory             1575.21k
      OS Memory          21688.00k
      Timestamp          22/04/2016 03:28:23 PM
      Step Count         115   Switch Count  35
      Page Faults        0
      Page Reclaims      357
      Page Swaps         0
      Voluntary Context Switches 144
      Involuntary Context Switches 20
      Block Input Operations 864
      Block Output Operations 264
```

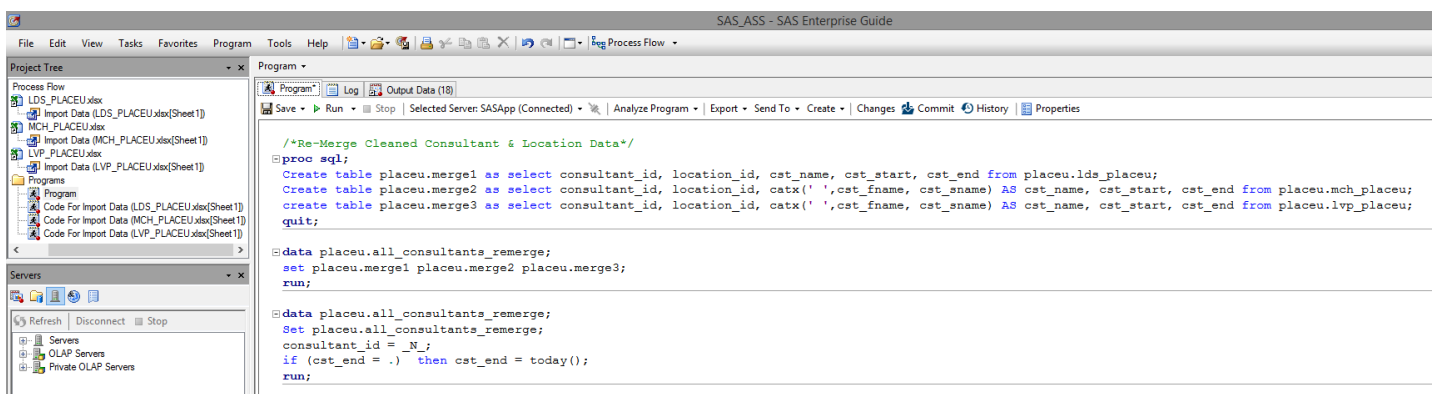


A data step was then used to merge the cleaned data and then load this data into the location dimension. A screenshot of the output can be found below.

The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left displays a process flow with three input data steps: LDS\_PLACEU.xlsx, MCH\_PLACEU.xlsx, and LVP\_PLACEU.xlsx. The main window shows the output of a data step, displaying a table with 3 rows and 3 columns: LOCATION\_ID, LOCATION\_NAME, and LOCATION\_POST.

	LOCATION_ID	LOCATION_NAME	LOCATION_POST
1	1	Leeds	L1 4HR
2	2	Manch	M11 3F
3	3	Liver	L1 4HR

The next stage was to perform another clean on the three data sources to include not only the consultant data, but also the location id so we can identify which consultant works at which location when it comes to the fact table, the newly created table is called 'all consultants remerged' and the code can be found below.



SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
    - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Log Output Data (18)

Export Send To Create Log Summary Project Log Properties

```

146      /*Re-Merge Cleaned Consultant & Location Data*/
147      proc sql;
148      Create table placeu.merge1 as select consultant_id, location_id, cst_name, cst_start, cst_end from placeu.lds_placeu;
149      NOTE: Table PLACEU.MERGE1 created, with 11 rows and 5 columns.
150
151      Create table placeu.merge2 as select consultant_id, location_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start,
152      ! cst_end from placeu.mch_placeu;
153      NOTE: Table PLACEU.MERGE2 created, with 11 rows and 5 columns.
154
155      create table placeu.merge3 as select consultant_id, location_id, catx(' ',cst_fname, cst_sname) AS cst_name, cst_start,
156      ! cst_end from placeu.lvp_placeu;
157      NOTE: Table PLACEU.MERGE3 created, with 11 rows and 5 columns.
158
159      quit;
160      NOTE: PROCEDURE SQL used (Total process time):
161      real time          0.04 seconds
162      user cpu time      0.01 seconds
163      system cpu time    0.00 seconds
164      memory             5506.75k
165      OS Memory         26032.00k
166      Timestamp         22/04/2016 03:28:23 PM
167      Step Count        117   Switch Count   51
168      Page Faults       0
169      Page Reclaims     227
170      Page Swaps        0
171      Voluntary Context Switches 248
172      Involuntary Context Switches 15
173      Block Input Operations 0
174      Block Output Operations 792
  
```

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
    - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Log Output Data (18)

Export Send To Create Log Summary Project Log Properties

```

153      data placeu.all_consultants_remerge;
154      set placeu.merge1 placeu.merge2 placeu.merge3;
155      run;
156
157      NOTE: There were 11 observations read from the data set PLACEU.MERGE1.
158      NOTE: There were 11 observations read from the data set PLACEU.MERGE2.
159      NOTE: There were 11 observations read from the data set PLACEU.MERGE3.
160      NOTE: The data set PLACEU.ALL_CONSULTANTS_REMERGE has 33 observations and 5 variables.
161      NOTE: DATA statement used (Total process time):
162      real time          0.02 seconds
163      user cpu time      0.00 seconds
164      system cpu time    0.00 seconds
165      memory             1577.06k
166      OS Memory         21688.00k
167      Timestamp         22/04/2016 03:28:23 PM
168      Step Count        118   Switch Count   35
169      Page Faults       0
170      Page Reclaims     350
171      Page Swaps        0
172      Voluntary Context Switches 149
173      Involuntary Context Switches 20
174      Block Input Operations 864
175      Block Output Operations 264
  
```

8 The SAS System 15:22 Friday, April 22, 2016

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
    - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
    - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Log Output Data (18)

Export Send To Create Log Summary Project Log Properties

```

157      data placeu.all_consultants_remerge;
158      Set placeu.all_consultants_remerge;
159      consultant_id = _N_;
160      if (cst_end = .) then cst_end = today();
161      run;
162
163      NOTE: There were 33 observations read from the data set PLACEU.ALL_CONSULTANTS_REMERGE.
164      NOTE: The data set PLACEU.ALL_CONSULTANTS_REMERGE has 33 observations and 5 variables.
165      NOTE: DATA statement used (Total process time):
166      real time          0.01 seconds
167      user cpu time      0.00 seconds
168      system cpu time    0.00 seconds
169      memory             1057.31k
170      OS Memory         21168.00k
171      Timestamp         22/04/2016 03:28:23 PM
172      Step Count        119   Switch Count   39
173      Page Faults       0
174      Page Reclaims     290
175      Page Swaps        0
176      Voluntary Context Switches 138
177      Involuntary Context Switches 10
178      Block Input Operations 288
179      Block Output Operations 264
  
```

The fact staging area can now be created.

The screenshot displays the SAS Enterprise Guide interface. On the left, the 'Servers' pane shows a tree structure with 'Servers', 'OLAP Servers', and 'Private OLAP Servers'. The main window is titled 'SAS\_ASS - SAS Enterprise Guide' and shows a 'Program' window with the following SQL code:

```
/*Reward Fact Staging Area*/
proc sql;
CREATE TABLE placeu.fact_stage
(
    consultant_id INTEGER,
    location_id INTEGER,
    time_id INTEGER,
    cst_start DATE,
    cst_start_year INTEGER,
    cst_end DATE,
    days INTEGER,
    months INTEGER,
    years INTEGER
);

insert into placeu.fact_stage select dc.consultant_id, ld.location_id, dt.time_id, dc.cst_start, year(dc.cst_start) AS cst_start_year,
dc.cst_end, datdif(dc.cst_start, dc.cst_end, 'act/act')AS days, INTCK('MONTH', dc.cst_start, dc.cst_end, 'C') AS months,
INTCK('YEAR', dc.cst_start, dc.cst_end, 'C') AS years
from placeu.dim_consultant dc, placeu.all_consultants_remerge acr, placeu.dim_time dt, placeu.dim_location ld
WHERE dc.consultant_id = acr.consultant_id
AND ld.location_id = acr.location_id
AND dt.year = year(dc.cst_start);
quit;
```

The 'Program' window also shows the execution results, including a note that the table 'PLACEU.FACT\_STAGE' was created with 0 rows and 9 columns, and that 33 rows were inserted into the table. The 'NOTE: PROCEDURE SQL used (Total process time):' section shows a real time of 0.03 seconds.

The above code creates a fact stage table and gathers the consultant id, consultant start date and consultant end date from the consultant dimension, the location id from the location dimension and the time id from the time dimension.

The measures for the fact table are also populated in this table with the use of the 'year()' function to just take the year from the consultant start date, the 'datdif' function to determine the number of days the consultant has worked for the company, and the 'intck' function to determine the number of months and years the consultant has worked for the company.

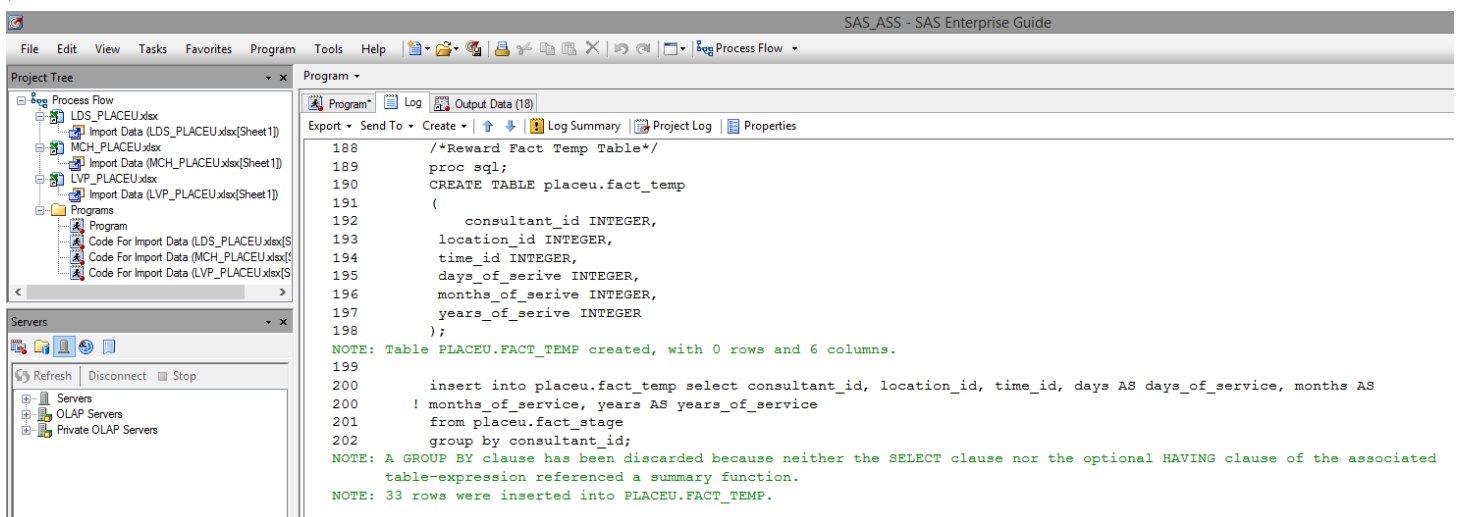
A join is then made on the consultant id from the consultant dimension and the consultant id from the all consultants remerged table, along with two other joins on the time dimension year and consultant start date with the use of the 'year()' function, and on the location id from the location dimension and the location id from the all consultants remerged table.

```
/*Reward Fact Temp Table*/
proc sql;
CREATE TABLE placeu.fact_temp
(
    consultant_id INTEGER,
    location_id INTEGER,
    time_id INTEGER,
    days_of_serive INTEGER,
    months_of_serive INTEGER,
    years_of_serive INTEGER
);

insert into placeu.fact_temp select consultant_id, location_id, time_id, days AS days_of_service, months AS months_of_service, years AS years_of_service
from placeu.fact_stage
group by consultant_id;

quit;

/*Create Reward Fact Sequence & Load Into Reward Fact*/
data placeu.FACT_reward;
Set placeu.fact_temp;
reward_id = _N_;
run;
```



A temporary fact table was then created which gathered the required values from the fact stage table and a data step was used to populate a sequence for the reward id and then load the data from the temporary table into the reward fact table. Ideally the data would be grouped by the reward id but due to this value not being in the temporary fact table and it being populated by a sequence SAS would not allow me to do this, so the decision was made to leave the rewards fact table being grouped by consultant id for the time being. The output and log for the reward fact can be found below.



SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Export • Send To • Create • Log • Output Data (17)

205

206 /\*Create Reward Fact Sequence & Load Into Reward Fact\*/

207

208 data placeu.FACT\_reward;

209 Set placeu.fact\_temp;

210 reward\_id = \_N\_;

211 run;

NOTE: There were 33 observations read from the data set PLACEU.FACT\_TEMP.

NOTE: The data set PLACEU.FACT\_REWARD has 33 observations and 7 variables.

NOTE: DATA statement used (Total process time):

real time	0.02 seconds
user cpu time	0.00 seconds
system cpu time	0.00 seconds
memory	1047.03k
OS Memory	21424.00k
Timestamp	22/04/2016 03:28:23 PM
Step Count	122 Switch Count 39
Page Faults	0
Page Reclaims	300
Page Swaps	0
Voluntary Context Switches	141
Involuntary Context Switches	6
Block Input Operations	32
Block Output Operations	264

211

212

213 \$ \_eg\_hidenotesandsource;

225

226

227 \$ \_eg\_hidenotesandsource;

230

15:22 Friday, April 22, 2016

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

- Servers
- OLAP Servers
- Private OLAP Servers

Program

Export • Send To • Create • Log • Output Data (18)

FACT\_REWARD

Filter and Sort Query Builder Where Data Describe Graph Analyze Export Send To

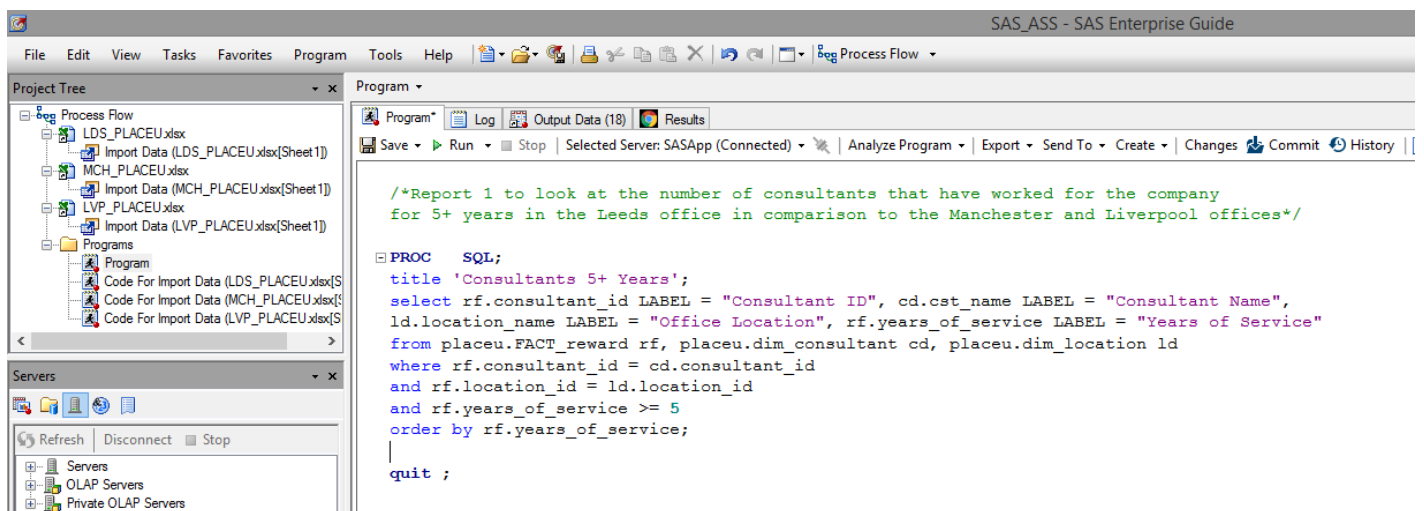
	consultant_id	location_id	time_id	days_of_service	months_of_service	years_of_service	reward_id
1	1	1	4	4688	154	12	1
2	2	1	15	701	23	1	2
3	3	1	10	2496	82	6	3
4	4	1	1	5910	194	16	4
5	5	1	13	1408	46	3	5
6	6	1	11	2131	70	5	6
7	7	1	12	1659	54	4	7
8	8	1	13	1400	46	3	8
9	9	1	10	2647	87	7	9
10	10	1	1	5783	190	15	10
11	11	1	16	368	12	1	11
12	12	2	4	4688	154	12	12
13	13	2	15	701	23	1	13
14	14	2	10	2496	82	6	14
15	15	2	1	1983	65	5	15
16	16	2	13	1408	46	3	16
17	17	2	11	2131	70	5	17
18	18	2	12	1659	54	4	18
19	19	2	13	1400	46	3	19
20	20	2	10	2647	87	7	20
21	21	2	1	1095	36	3	21
22	22	2	16	368	12	1	22
23	23	3	4	4688	154	12	23
24	24	3	15	701	23	1	24
25	25	3	10	2496	82	6	25
26	26	3	1	5910	194	16	26
27	27	3	13	1408	46	3	27
28	28	3	11	2131	70	5	28
29	29	3	12	1659	54	4	29
30	30	3	13	1400	46	3	30
31	31	3	10	2647	87	7	31
32	32	3	1	5783	190	15	32
33	33	3	16	368	12	1	33



### Article III. Data Analysis

Due to the MD's objectives which were defined in part 1 (data warehouse design approach assignment) a number of report have been produced in order to allow him to issue rewards to consultant.

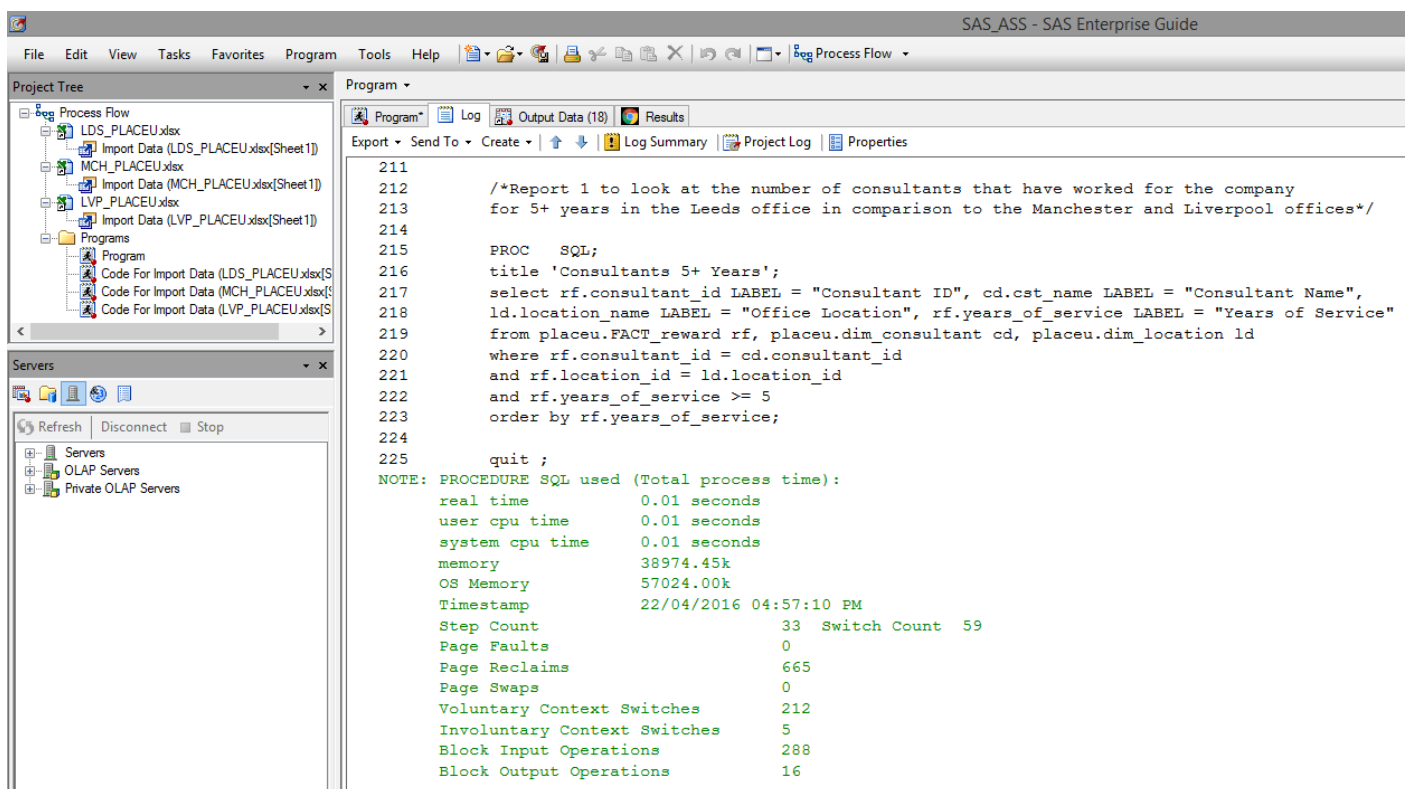
The first report looked at the number of consultants that have worked for the company for 5 or more years in the Leeds office in comparison to the Manchester and Liverpool offices. The code, log and results can be found below.



The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left displays a process flow with three import data steps (LDS\_PLACEU.xlsx, MCH\_PLACEU.xlsx, LVP\_PLACEU.xlsx) and a program step. The Program window on the right contains the following SAS code:

```
/*Report 1 to look at the number of consultants that have worked for the company
for 5+ years in the Leeds office in comparison to the Manchester and Liverpool offices*/

PROC SQL;
title 'Consultants 5+ Years';
select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL = "Consultant Name",
ld.location_name LABEL = "Office Location", rf.years_of_service LABEL = "Years of Service"
from placeu.FACT_reward rf, placeu.dim_consultant cd, placeu.dim_location ld
where rf.consultant_id = cd.consultant_id
and rf.location_id = ld.location_id
and rf.years_of_service >= 5
order by rf.years_of_service;
quit ;
```



The screenshot shows the SAS Enterprise Guide interface with the same project flow. The Program window now displays the SAS log output for the executed program:

```
211
212 /*Report 1 to look at the number of consultants that have worked for the company
213 for 5+ years in the Leeds office in comparison to the Manchester and Liverpool offices*/
214
215 PROC SQL;
216 title 'Consultants 5+ Years';
217 select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL = "Consultant Name",
218 ld.location_name LABEL = "Office Location", rf.years_of_service LABEL = "Years of Service"
219 from placeu.FACT_reward rf, placeu.dim_consultant cd, placeu.dim_location ld
220 where rf.consultant_id = cd.consultant_id
221 and rf.location_id = ld.location_id
222 and rf.years_of_service >= 5
223 order by rf.years_of_service;
224
225 quit ;
NOTE: PROCEDURE SQL used (Total process time):
      real time           0.01 seconds
      user cpu time       0.01 seconds
      system cpu time     0.01 seconds
      memory              38974.45k
      OS Memory           57024.00k
      Timestamp           22/04/2016 04:57:10 PM
      Step Count          33      Switch Count  59
      Page Faults         0
      Page Reclaims       665
      Page Swaps          0
      Voluntary Context Switches 212
      Involuntary Context Switches 5
      Block Input Operations 288
      Block Output Operations 16
```

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Process Flow

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx[S
      - Code For Import Data (MCH\_PLACEU.xlsx[S
      - Code For Import Data (LVP\_PLACEU.xlsx[S

Servers

Refresh Disconnect Stop

Servers OLAP Servers Private OLAP Servers

Program

Program\* Log Output Data (18) Results

Refresh Export Send To Publish Properties

Consultants 5+ Years

Consultant ID	Consultant Name	Office Location	Years of Service
28	Freya Fell	Liver	5
15	Danny Dared	Manch	5
17	Freddie Frumped	Manch	5
6	Fred Frump	Leeds	5
3	Charlie Crumble	Leeds	6
14	Cleo Crumbled	Manch	6
25	Champ Crim	Liver	6
9	Izzac Ingle	Leeds	7
31	Lilly Lolly	Liver	7
20	Imogen Ingleby	Manch	7
12	Jake Abbot	Manch	12
23	Abbie Abet	Liver	12
1	Aaron Abbots	Leeds	12
32	Julia Juke	Liver	15
10	Jenna Jenkin	Leeds	15
4	Dan Dare	Leeds	16
26	Drew Drake	Liver	16

Looking at all the consultants from all locations and their years of service a graph has been produced for easy analysis. The code, log and results can be found below.

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Process Flow

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx[S
      - Code For Import Data (MCH\_PLACEU.xlsx[S
      - Code For Import Data (LVP\_PLACEU.xlsx[S

Servers

Refresh Disconnect Stop

Servers OLAP Servers Private OLAP Servers

Program

Program\* Log Output Data (19) Results

Save Run Stop Selected Server: SASApp (Connected) Analyze Program Export Send To Create

```

/* Set Graphics Environment */
options reset=all border cback=white htext=10pt htitle=12pt;

/* Create Data Set for Graph */

proc sql;
CREATE TABLE placeu.consultant_graph
(
    cst_name varchar (255),
    years_of_service INTEGER
);

insert into placeu.consultant_graph select c.cst_name, r.years_of_service
from placeu.FACT_reward r, placeu.dim_consultant c
where r.consultant_id = c.consultant_id;

quit;

/*Titles and Labels*/
axis1 label=('Consultants');
axis2 label=('Years Of Service');
title1 'Consultants Years of Service';

/* Create Graph */
proc gchart data=placeu.consultant_graph;
hbar cst_name/ maxis=axis1 raxis=axis2 nostats
sumvar=years_of_service;
run;

```

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx[Sheet1])
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx[Sheet1])
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx[Sheet1])
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (MCH\_PLACEU.xlsx[Sheet1])
      - Code For Import Data (LVP\_PLACEU.xlsx[Sheet1])

Servers

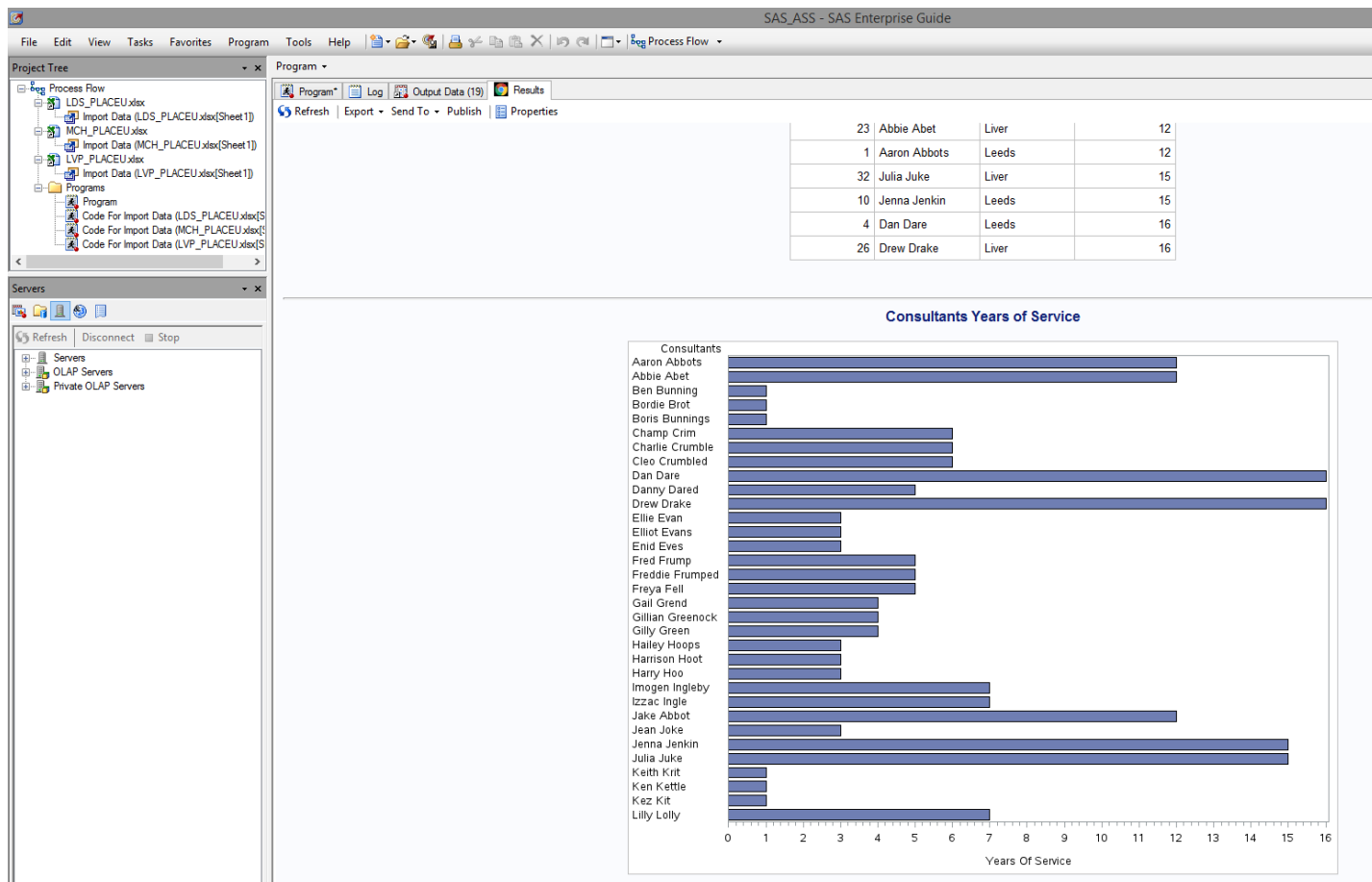
- Servers
- OLAP Servers
- Private OLAP Servers

Program

Export Send To Create Log Output Data (20) Results

```

227 goptions reset=all border cback=white htext=10pt htitle=12pt;
228                                The SAS System
229
230 /* Create Data Set for Graph */
231
232 proc sql;
233     CREATE TABLE placeu.consultant_graph
234     (
235         cst_name varchar (255),
236         years_of_service INTEGER
237     );
238 NOTE: Table PLACEU.CONCONSULTANT_GRAPH created, with 0 rows and 2 columns.
239
240 insert into placeu.consultant_graph select c.cst_name, r.years_of_service
241 from placeu.FACT_reward r, placeu.dim_consultant c
242 where r.consultant_id = c.consultant_id;
243 NOTE: 33 rows were inserted into PLACEU.CONCONSULTANT_GRAPH.
244
245 quit;
246 NOTE: PROCEDURE SQL used (Total process time):
247     real time           0.03 seconds
248     user cpu time       0.01 seconds
249     system cpu time     0.00 seconds
250     memory              6278.81k
251     OS Memory          28608.00k
252     Timestamp          28/04/2016 01:14:05 PM
253     Step Count         399    Switch Count   67
254     Page Faults        0
255     Page Reclaims     170
256     Page Swaps         0
257     Voluntary Context Switches 265
258     Involuntary Context Switches 14
259     Block Input Operations 288
260     Block Output Operations 528
261
262 /*Titles and Labels*/
263 axis1 label=('Consultants');
264 axis2 label=('Years Of Service');
265 title1 'Consultants Years of Service';
266 /* Create Graph */
267
268 proc gchart data=placeu.consultant_graph;
269     hbar cst_name/ maxis=axis1 raxis=axis2 nostats
270     sumvar=years_of_service;
271 run;
  
```



The next report that was produced was in order for the MD to view all the consultants that have started working for the company in the last year. The code, log and results can be found below.

The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left displays a process flow with steps for importing data from LDS\_PLACEU.xlsx, MCH\_PLACEU.xlsx, and LVP\_PLACEU.xlsx. The main window shows the SAS code for a PROC SQL query titled 'Consultants New Starters'. The code selects consultant IDs, names, office locations, and start dates from the placeu.FACT\_reward, placeu.dim\_consultant, and placeu.dim\_location tables, filtered by a start date between January 15, 2015, and the current date.

```

/*Report 2 to look at the number of consultants that have started working for the company over the last year*/
PROC SQL;
title 'Consultants New Starters';
select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL = "Consultant Name",
ld.location_name LABEL = "Office Location", cd.cst_start LABEL = "Start Date"
from placeu.FACT_reward rf, placeu.dim_consultant cd, placeu.dim_location ld
where rf.consultant_id = cd.consultant_id
and rf.location_id = ld.location_id
and cd.cst_start between "01jan15"d and today()
order by rf.consultant_id;
quit ;

```

The screenshot shows the SAS Enterprise Guide interface with the execution log and results for the PROC SQL query. The log displays the PROC SQL statement and a summary of the procedure's performance, including real time, user CPU time, system CPU time, memory usage, and timestamps. The results window shows the output of the query, which is a list of consultants who started working in the last year.

```

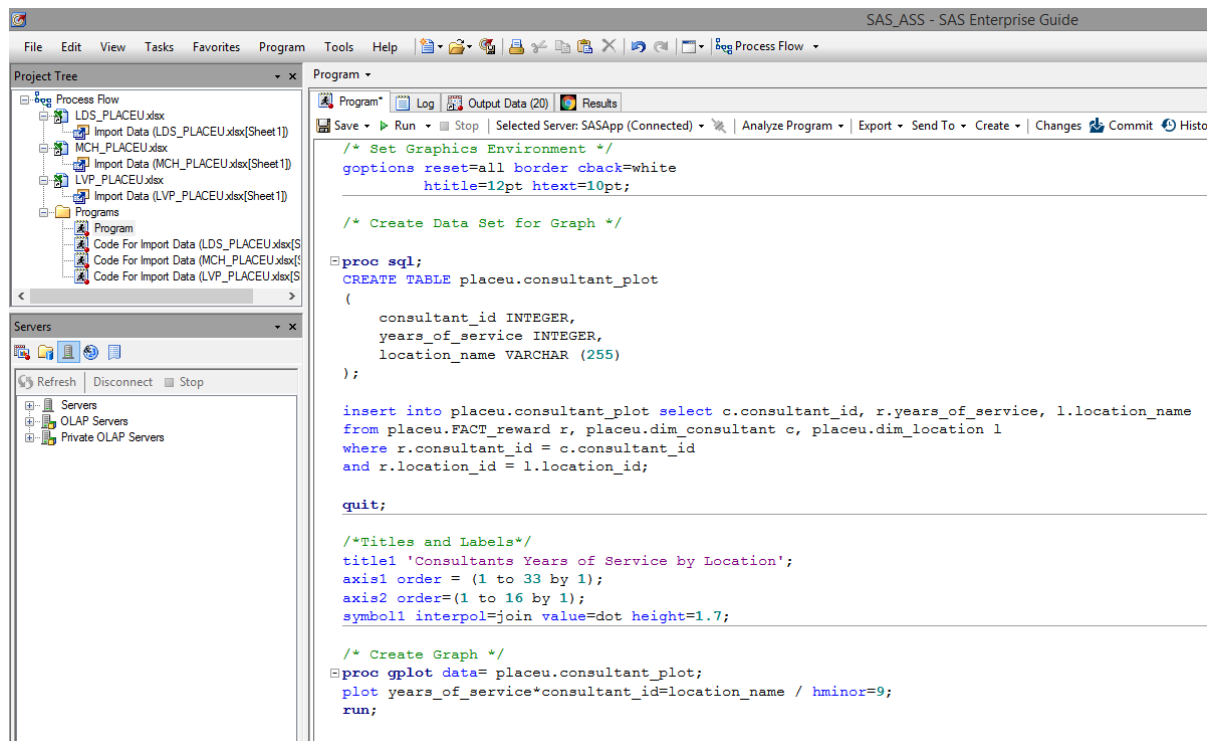
257 PROC SQL;
258
259 title 'Consultants New Starters';
260 select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL = "Consultant Name",
261 ld.location_name LABEL = "Office Location", cd.cst_start LABEL = "Start Date"
262 from placeu.FACT_reward rf, placeu.dim_consultant cd, placeu.dim_location ld
263 where rf.consultant_id = cd.consultant_id
264 and rf.location_id = ld.location_id
265 and cd.cst_start between "01jan15"d and today()
266 order by rf.consultant_id;
267 quit ;
NOTE: PROCEDURE SQL used (Total process time):
real time 0.01 seconds
user cpu time 0.01 seconds
system cpu time 0.00 seconds
memory 38223.60k
OS Memory 58304.00k
Timestamp 28/04/2016 11:39:03 AM
Step Count 77 Switch Count 53
Page Faults 0
Page Reclaims 413
Page Swaps 0
Voluntary Context Switches 180
Involuntary Context Switches 0
Block Input Operations 0
Block Output Operations 0

```

### Consultants New Starters

Consultant ID	Consultant Name	Office Location	Start Date
11	Ken Kettle	Leeds	20APR15
22	Kez Kit	Manch	20APR15
33	Keith Krit	Liver	20APR15

Looking again at all the consultants from all locations and their years of service a graph has been produced for easy analysis. However, this graph has split the consultants up by the different locations. The code, log and results can be found below.



The screenshot shows the SAS Enterprise Guide interface. The Project Tree on the left displays a process flow with steps for importing data from LDS\_PLACEU.xlsx, MCH\_PLACEU.xlsx, and LVP\_PLACEU.xlsx. The main window shows the following SAS code:

```

/* Set Graphics Environment */
goptions reset=all border cback=white
        htitle=12pt htext=10pt;

/* Create Data Set for Graph */

proc sql;
CREATE TABLE placeu.consultant_plot
(
    consultant_id INTEGER,
    years_of_service INTEGER,
    location_name VARCHAR (255)
);

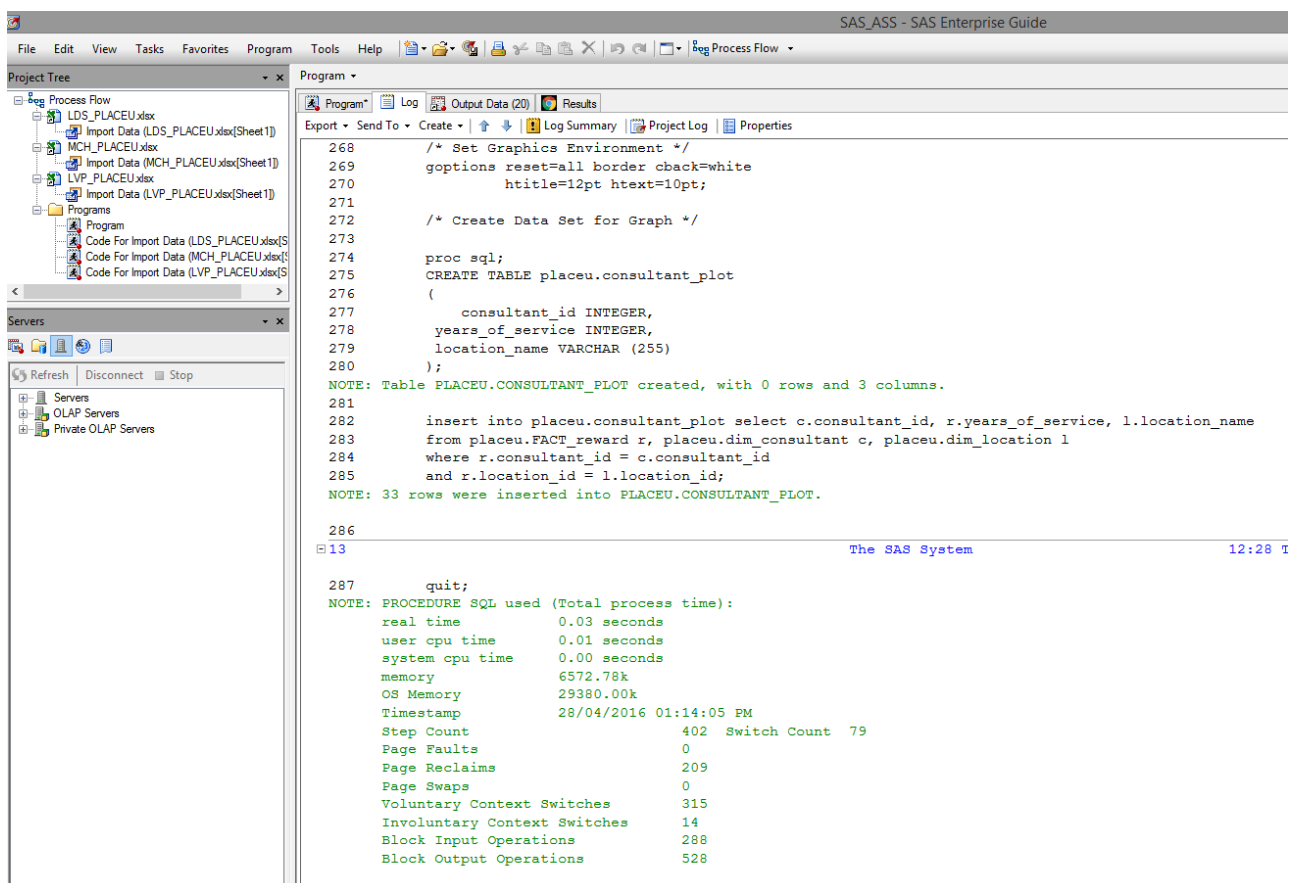
insert into placeu.consultant_plot select c.consultant_id, r.years_of_service, l.location_name
from placeu.FACT_reward r, placeu.dim_consultant c, placeu.dim_location l
where r.consultant_id = c.consultant_id
and r.location_id = l.location_id;

quit;

/*Titles and Labels*/
title1 'Consultants Years of Service by Location';
axis1 order = (1 to 33 by 1);
axis2 order=(1 to 16 by 1);
symbol1 interpol=join value=dot height=1.7;

/* Create Graph */
proc gplot data= placeu.consultant_plot;
plot years_of_service*consultant_id=location_name / hminor=9;
run;

```



The screenshot shows the SAS log output for the previous code. The log includes the following information:

```

268 /* Set Graphics Environment */
269 goptions reset=all border cback=white
270         htitle=12pt htext=10pt;
271
272 /* Create Data Set for Graph */
273
274 proc sql;
275 CREATE TABLE placeu.consultant_plot
276 (
277     consultant_id INTEGER,
278     years_of_service INTEGER,
279     location_name VARCHAR (255)
280 );
NOTE: Table PLACEU.CONCONSULTANT_PLOT created, with 0 rows and 3 columns.
281
282 insert into placeu.consultant_plot select c.consultant_id, r.years_of_service, l.location_name
283 from placeu.FACT_reward r, placeu.dim_consultant c, placeu.dim_location l
284 where r.consultant_id = c.consultant_id
285 and r.location_id = l.location_id;
NOTE: 33 rows were inserted into PLACEU.CONCONSULTANT_PLOT.

286
13 The SAS System 12:28 T

287 quit;
NOTE: PROCEDURE SQL used (Total process time):
      real time          0.03 seconds
      user cpu time      0.01 seconds
      system cpu time    0.00 seconds
      memory             6572.78k
      OS Memory          29380.00k
      Timestamp          28/04/2016 01:14:05 PM
      Step Count         402  Switch Count  79
      Page Faults        0
      Page Reclaims      209
      Page Swaps         0
      Voluntary Context Switches 315
      Involuntary Context Switches 14
      Block Input Operations 288
      Block Output Operations 528

```

SAS\_ASS - SAS Enterprise Guide

File Edit View Tasks Favorites Program Tools Help

Project Tree

- Process Flow
  - LDS\_PLACEU.xlsx
    - Import Data (LDS\_PLACEU.xlsx(Sheet1))
  - MCH\_PLACEU.xlsx
    - Import Data (MCH\_PLACEU.xlsx(Sheet1))
  - LVP\_PLACEU.xlsx
    - Import Data (LVP\_PLACEU.xlsx(Sheet1))
  - Programs
    - Program
      - Code For Import Data (LDS\_PLACEU.xlsx(Sheet1))
      - Code For Import Data (MCH\_PLACEU.xlsx(Sheet1))
      - Code For Import Data (LVP\_PLACEU.xlsx(Sheet1))

Servers

- Refresh Disconnect Stop
- Servers
  - OLAP Servers
  - Private OLAP Servers

Program

Log Output Data (20) Results

Export Send To Create Log Summary Project Log Properties

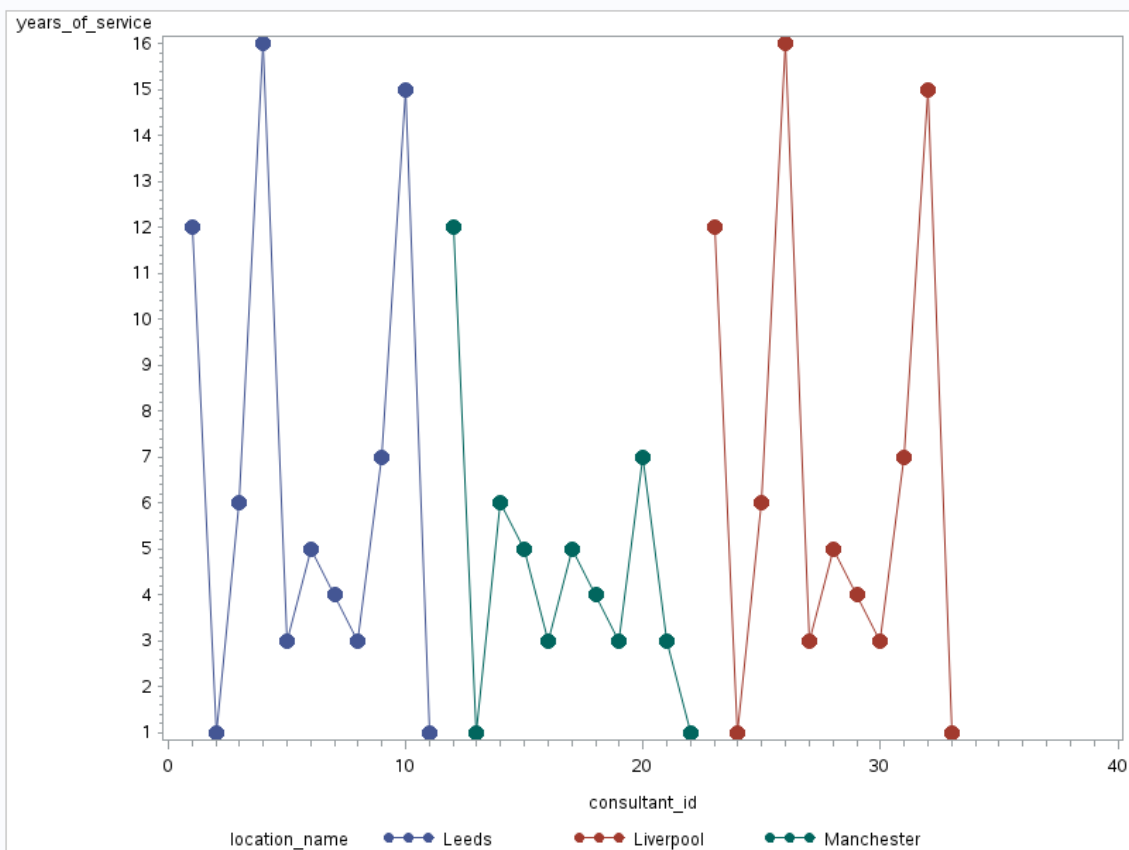
```

user cpu time      0.01 seconds
system cpu time    0.00 seconds
memory            6572.78k
OS Memory         29380.00k
Timestamp         28/04/2016 01:14:05 PM
Step Count        402  Switch Count  79
Page Faults       0
Page Reclaims     209
Page Swaps        0
Voluntary Context Switches 315
Involuntary Context Switches 14
Block Input Operations 288
Block Output Operations 528

288
289
290 /*Titles and Labels*/
291 title1 'Consultants Years of Service by Location';
292 axis1 order = (1 to 33 by 1);
293 axis2 order=(1 to 16 by 1);
294 symbol1 interpol=join value=dot height=1.7;
295
296 /* Create Graph */
297 proc gplot data= placeu.consultant_plot;
298 plot years_of_service*consultant_id=location_name / hminor=9;
299 run;
300
301 NOTE: 42426 bytes written to /saswork/SAS_workD138000118F6_odaws01-prod-ie/SAS_work9AFB000118F6_odaws01-prod-ie/gplot20.png.
302
303
304 %_eg_hidenotesandsource;
305
306
307 %_eg_hidenotesandsource;
308
309
310
311
312
313
314
315
316
317
318
319
320
321

```

Consultants Years of Service by Location

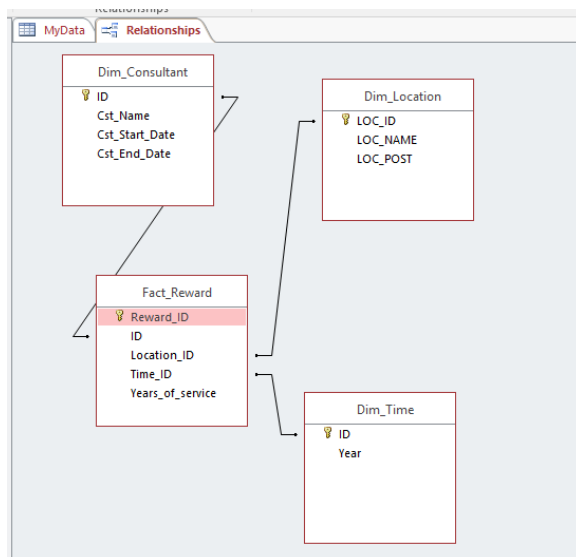


## Article IV. OLAP Using Excel

A detail table was loaded into Microsoft Access as you can see in the screenshot below. The dimensions and fact table was then created and populated with this data through a number of queries.

MyData										
ID	Consultant	Cst_Name	Cst_Start_Di	Cst_End_Da	Location_ID	Loc_Name	Loc_Post	Time_ID	Years_of_se	Click to Add
1	Aaron Abbotts	6/22/2003			1 Leeds	LS8 4HT		4	13	
2	Ben Bunning	5/22/2014			1 Leeds	LS8 4HT		15	2	
3	Charlie Crumb	6/22/2009			1 Leeds	LS8 4HT		10	7	
4	Dan Dare	2/16/2000			1 Leeds	LS8 4HT		1	16	
5	Elliot Evans	6/14/2012			1 Leeds	LS8 4HT		13	4	
6	Fred Frump	6/22/2010			1 Leeds	LS8 4HT		11	6	
7	Gilly Green	07/10/2011			1 Leeds	LS8 4HT		12	5	
8	Harry Hoo	6/22/2012			1 Leeds	LS8 4HT		13	4	
9	Izzac Ingle	1/22/2009			1 Leeds	LS8 4HT		10	7	
10	Jenna Jenkin	6/22/2000			1 Leeds	LS8 4HT		1	16	
11	Ken Kettle	4/20/2015			1 Leeds	LS8 4HT		16	1	
12	Jake Abbot	6/22/2003			2 Manchester	M11 3FF		4	13	
13	Boris Bunnings	5/22/2014			2 Manchester	M11 3FF		15	2	
14	Cleo Crumblec	6/22/2009			2 Manchester	M11 3FF		10	7	
15	Danny Dared	2/16/2000	6/22/2005		2 Manchester	M11 3FF		1	16	
16	Ellie Evan	6/14/2012			2 Manchester	M11 3FF		13	4	
17	Freddie Frump	6/22/2010			2 Manchester	M11 3FF		11	6	
18	Gillian Greeno	07/10/2011			2 Manchester	M11 3FF		12	5	
19	Harrison Hoot	6/22/2012			2 Manchester	M11 3FF		13	4	
20	Imogen Ingleb	1/22/2009	6/22/2010		2 Manchester	M11 3FF		10	7	
21	Jean Joke	6/22/2000	6/22/2011		2 Manchester	M11 3FF		1	16	
22	Kez Kit	4/20/2015			2 Manchester	M11 3FF		16	1	
23	Abbie Abet	6/22/2003			3 Liverpool	L1 4HR		4	13	
24	Bordie Brot	5/22/2014			3 Liverpool	L1 4HR		15	2	
25	Champ Crim	6/22/2009			3 Liverpool	L1 4HR		10	7	
26	Drew Drake	2/16/2000			3 Liverpool	L1 4HR		1	16	
27	Enid Eve	6/14/2012			3 Liverpool	L1 4HR		13	4	
28	Freya Fell	6/22/2010			3 Liverpool	L1 4HR		11	6	
29	Gail Grend	07/10/2011			3 Liverpool	L1 4HR		12	5	
30	Hailey Hoops	6/22/2012			3 Liverpool	L1 4HR		13	4	
31	Iilly Iolly	1/22/2009			3 Liverpool	L1 4HR		10	7	
32	Julia Juke	6/22/2000			3 Liverpool	L1 4HR		1	16	
33	Keith Krit	4/20/2015			3 Liverpool	L1 4HR		16	1	
*(New)								0	0	

Once the dimensions and fact table was created, relationships were formed like you can see in the screenshot below.



The next stage was to create a query which would include relevant data required for online analytical processing (OLAP) using Microsoft Excel. Below you can see the results from the query.

Cst_Name	Cst_Start_Dt	Cst_End_Da	LOC_NAME	Year	Years_of_se
Aaron Abbots	6/22/2003		Leeds	2003	13
Ben Bunning	5/22/2014		Leeds	2014	2
Charlie Crumb	6/22/2009		Leeds	2009	7
Dan Dare	2/16/2000		Leeds	2000	16
Elliot Evans	6/14/2012		Leeds	2012	4
Fred Frump	6/22/2010		Leeds	2010	6
Gilly Green	07/10/2011		Leeds	2011	5
Harry Hoo	6/22/2012		Leeds	2012	4
Izzac Ingle	1/22/2009		Leeds	2009	7
Jenna Jenkin	6/22/2000		Leeds	2000	16
Ken Kettle	4/20/2015		Leeds	2015	1
Jake Abbot	6/22/2003		Manchester	2003	13
Boris Bunnings	5/22/2014		Manchester	2014	2
Cleo Crumblec	6/22/2009		Manchester	2009	7
Danny Dared	2/16/2000	6/22/2005	Manchester	2000	16
Ellie Evan	6/14/2012		Manchester	2012	4
Freddie Frump	6/22/2010		Manchester	2010	6
Gillian Greeno	07/10/2011		Manchester	2011	5
Harrison Hoot	6/22/2012		Manchester	2012	4
Imogen Ingleb	1/22/2009	6/22/2010	Manchester	2009	7
Jean Joke	6/22/2000	6/22/2011	Manchester	2000	16
Kez Kit	4/20/2015		Manchester	2015	1
Abbie Abet	6/22/2003		Liverpool	2003	13
Bordie Brot	5/22/2014		Liverpool	2014	2
Champ Crim	6/22/2009		Liverpool	2009	7
Drew Drake	2/16/2000		Liverpool	2000	16
Enid Eve	6/14/2012		Liverpool	2012	4
Freya Fell	6/22/2010		Liverpool	2010	6
Gail Grend	07/10/2011		Liverpool	2011	5
Hailey Hoops	6/22/2012		Liverpool	2012	4
lilly lolly	1/22/2009		Liverpool	2009	7
Julia Juke	6/22/2000		Liverpool	2000	16
Keith Krit	4/20/2015		Liverpool	2015	1

This query data was then loaded into Excel and a pivot table was implemented to demonstrate 'drill up' and 'drill down'. Below you can see the pivot table that was created which shows the consultants categorised by each location, along with their years of service.

Cst_End_Date	(All)				
Row Labels	Sum of Years_of_service				
<b>Leeds</b>	<b>81</b>				
Aaron Abbots	13				
Ben Bunning	2				
Charlie Crumble	7				
Dan Dare	16				
Elliot Evans	4				
Fred Frump	6				
Gilly Green	5				
Harry Hoo	4				
Izzac Ingle	7				
Jenna Jenkin	16				
Ken Kettle	1				
<b>Liverpool</b>	<b>81</b>				
Abbie Abet	13				
Bordie Brot	2				
Champ Crim	7				
Drew Drake	16				
Enid Eve	4				
Freya Fell	6				
Gail Grend	5				
Hailey Hoops	4				
Julia Juke	16				
Keith Krit	1				
lilly lolly	7				
<b>Manchester</b>	<b>81</b>				
Boris Bunnings	2				
Cleo Crumbled	7				
Danny Dared	16				
Ellie Evan	4				
Freddie Frumpec	6				
Gillian Greenock	5				
Harrison Hoot	4				
Imogen Ingleby	7				
Jake Abbot	13				
Jean Joke	16				
Kez Kit	1				
<b>Grand Total</b>	<b>243</b>				



A filter was applied which determines the consultants who no longer work for the company, this can be used to determine how many consultants no longer work for the company, how many years of service they did provide and what date they left.

Sliders was also used as a quick and easy tool to control the values shown, for example if you wanted to determine any new starters you could click the number 1 under the years of service tab which will pull up all consultants who have worked at the company for 1 year, the same could be done if wanting to determine the consultants who had worked at the company for 5 years, you would just click the number 5 under the years of service tab.

## **Article V. Findings & Evaluation**

Above the results from the final data warehouse implementation can be seen. However due to the MD's requirements that were captured during the planning stage along with his key defining business objectives, a scaled-down data warehouse known as a data mart was implemented in order to meet his demands and provide fast results with an outlook towards future expansion. (Hammergren and Simon, 2009)

This approach is successfully used and suggested, due to it having a focus on one ODS at a time, (Chenoweth, Corral and Demirkan, 2006) and along with it being known for its speed, it is also a cheap solution with less complexity and risk than a full scale data warehouse. (Hammergren and Simon, 2009)

The same methodology that was initially outlined in the planning stage was used along with the ETL process, but just on a smaller scale than initially intended. The approach chosen was Ralph Kimball's bottom-up approach which uses data marts as a starting point with the intention of generating multiple marts and then combining them to develop a data warehouse, at the moment the MD can only produce reports and queries on his consultants that work across the three branches, however a new data mart could be produced on his accounts and his contractors, or he could simply use the existing mart and add new dimensions. (Standen, 2008)

The star schema technique which was developed by Ralph Kimball, and can be found above in section 2.02 uses a denormalized structure made up of dimension tables and a fact table in order for the MD to integrate reward schemes within his company. (Hammergren and Simon, 2009)

As briefly mentioned above, the decision to use a surrogate key for the fact table was made in order to speed up the process of data retrieval and allow the data warehouse to store historic data. A surrogate key has been assigned in SAS with the use of a sequence, so although this ID has no meaning, it is always numeric and each time a new row is added, the program will automatically generate the next number in a sequence and assign this to the Reward ID. (Kimball, 1998)

As discussed in the planning phase the transformation stage in the ETL process was used to find any inconsistencies, duplicates or missing data. Staging areas was used in the data gap analysis in order to avoid damaging the raw data along with the dimensions and fact table, and SQL was used on attributes with missing data such as the consultant end date – this may be due to the fact that these consultants still work for the company, however to avoid null values in the queries and reports, today's date was generated and imputed in this field if it was missing.

Transformation of the data is a process to clean the data in order to make it more meaningful when it comes to using the data to perform better business decisions. The most common field that requires data cleansing is the name and address field, and this was also the case with PlaceU. (Ballard et al, 1998)

It was important to join the consultants first name along with their last name as the Manchester and Liverpool branches had two separate columns whereas the Leeds branch had these two fields already combined. The decision to split the Leeds name column into two fields could have been made, however this would have required more complexity and therefore a longer timescale, and although the consultants name is required, a split was not necessary so the decision to concatenate was made.

The use of queries and report analysis is used to answer questions and relay this information in a simplified, readable format. (Ballard and others, 1998) the questions being, how many consultants have worked for the company for 5 or more years in the Leeds office in comparison to the Manchester and Liverpool offices? How many consultant have worked for the company for 10 or more years in the Leeds office in comparison to the Manchester and Liverpool offices? And how many consultants have started working for the company in the last year?

So does the data warehouse fulfil the MD's initial requirements? And are the queries and reports that have been produced useful for the MD? Yes, the MD can easily see how many consultants have worked for the company for 5 or more years in each of his three branches so he can compare statistics and reward his loyal consultants, he can also see all new starters, which he may want to offer benefit packages to.

Although the query to determine consultants that have worked for the company for 10 or more years was not demonstrated above, this could easily be accomplished by changing the number 5 to the number 10 in the first query.

In order to show a range of outputs and allow the MD to answer the question, how many consultants have worked for the company for 10 or more years in each of his three branches? Two graphs were produced, one took a lead focus on the consultants themselves with their name and years of service outputted and the second took a lead on the location which would give the MD a visual way to detect which location has the most loyal staff.

Thinking about the on-going maintenance, it is important to remember that data warehouses are rarely updated or deleted from, usually new data is just inserted at certain time intervals. In PlaceU's case the initial data load is now complete and due to the MD wanting to look at figures on a yearly basis, it will be another financial year before new data will be added.

A data warehouse is a useful tool to track history and attributes can be either historically significant or historically insignificant, in PlaceU's case, a slowly changing dimension type 1 should be used due to the MD's objectives making it not necessary

to keep any historical data, all records will be overwritten if changes are made, however if for example the MD changed his objectives or requirements at a later date, which many people do, all historic data would be lost which is why a type 2 slowly changing dimension has been implemented, although this does mean that the data will grow quickly and the query's and reports will need to be slightly changed to avoid duplicates affecting the overall business results. (Ross, 2015)

In article 4 OLAP; online analytical processing has been used with excel in order to perform ad hoc analysis on PlaceU's data in order to provide an insight into the company and aid in the decision making process. This is a quicker and simpler way of meeting the MD's business objectives and is widely used in the industry. (OLAP, 2016)

In order for data warehouses to become useful it is important that both developers, users and managers all think systematically and determine the difference between what is required and what is desired. At the moment there is not anything else the MD required, however future plans, apart from expanding the mart into a fully-fledged data warehouse, may include a comparison with competitor's data which would need to be sourced online, purchased directly from a company or acquired from market research, and along with this, comes a whole new set of complications, but that is for another day. (Ballou and Tayi, 1999)

## Article VI. Bibliography

Ballard, C, et all (1998) **Data Modeling Techniques for Data Warehousing**. 1<sup>st</sup> Ed. California, International Business Machines Corporation.

Ballou, D.P and Tayi, G.K (1999) Enhancing data quality in data warehouse environments. **Communications of the ACM**, 42 (1) January, pp.73-78.

Chenoweth, T, Corral, K and Demirkan, H (2006) **Seven Key Interventions for Data Warehouse Success**. [Online] Available from:  
<<http://dss.gusconstan.com/DSS/documents/p114-chenoweth.pdf>> [Accessed 2<sup>nd</sup> May 2016].

Hammergren, T.C and Simon, A.R (2009) **Data Warehousing for Dummies**. 2<sup>nd</sup> Ed. Indiana, Wiley Publishing.

Kimball, R (1998) **Surrogate Keys**. [Online] Available from:  
<<http://www.kimballgroup.com/1998/05/surrogate-keys/>> [Accessed 4<sup>th</sup> may 2016].

OLAP (2016) **What is the definition of OLAP?** [Online] Available from:  
<<http://olap.com/olap-definition/>> [Accessed 4<sup>th</sup> Mary 2016].

Ross, M (2015) **Slowly Changing Dimensions Are Not Always as Easy as 1, 2, 3**. [Online] Available from: < <http://www.kimballgroup.com/2005/03/slowly-changing-dimensions-are-not-always-as-easy-as-1-2-3/> > [Accessed 4<sup>th</sup> May 2016].

Standen, J (2008) **Data Warehouse vs. Data Mart**. [Online] Available from:  
<<http://www.datamartist.com/data-warehouse-vs-data-mart>> [Accessed 2<sup>nd</sup> May 2016].

## Article VII. Appendix

### Section 7.01 SAS Code

```
libname placeu "/home/j.kennedy57580/sasuser.v94/placeu";

/*Create Tables*/
proc sql;
CREATE TABLE placeu.dim_location(
    location_id INTEGER NOT NULL,
    location_name VARCHAR (255),
    location_post VARCHAR (10),
    CONSTRAINT pk_location PRIMARY KEY (location_id)
);

CREATE TABLE placeu.dim_time(
    time_id INTEGER NOT NULL,
    year INTEGER,
    CONSTRAINT pk_time PRIMARY KEY (time_id)
);

CREATE TABLE placeu.dim_consultant(
    consultant_id INTEGER NOT NULL,
    cst_name VARCHAR (255),
    cst_start DATE,
    cst_end DATE,
    CONSTRAINT pk_con PRIMARY KEY (consultant_id)
);

CREATE TABLE placeu.FACT_reward(
    reward_id INTEGER NOT NULL,
    time_id INTEGER NOT NULL,
    location_id INTEGER NOT NULL,
    consultant_id INTEGER NOT NULL,
    days_of_service INTEGER,
    months_of_service INTEGER,
    years_of_service INTEGER,
    CONSTRAINT pk_fact PRIMARY KEY (reward_id)
);
quit;

/*Manual Insert Into Time Dimension*/
proc sql;
INSERT INTO placeu.dim_time
VALUES (1, 2000);
INSERT INTO placeu.dim_time
VALUES (2, 2001);
INSERT INTO placeu.dim_time
VALUES (3, 2002);
INSERT INTO placeu.dim_time
VALUES (4, 2003);
INSERT INTO placeu.dim_time
VALUES (5, 2004);
```

```

INSERT INTO placeu.dim_time
VALUES (6, 2005);
INSERT INTO placeu.dim_time
VALUES (7, 2006);
INSERT INTO placeu.dim_time
VALUES (8, 2007);
INSERT INTO placeu.dim_time
VALUES (9, 2008);
INSERT INTO placeu.dim_time
VALUES (10, 2009);
INSERT INTO placeu.dim_time
VALUES (11, 2010);
INSERT INTO placeu.dim_time
VALUES (12, 2011);
INSERT INTO placeu.dim_time
VALUES (13, 2012);
INSERT INTO placeu.dim_time
VALUES (14, 2013);
INSERT INTO placeu.dim_time
VALUES (15, 2014);
INSERT INTO placeu.dim_time
VALUES (16, 2015);
INSERT INTO placeu.dim_time
VALUES (17, 2016);
quit;

/*Consultant Staging Area*/
proc sql;
Create table placeu.stagearea as select consultant_id, cst_name,
cst_start, cst_end from placeu.lds_placeu;
Create table placeu.stageareaMan as select consultant_id, catx('
',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from
placeu.mch_placeu;
create table placeu.stageareaLiv as select consultant_id, catx('
',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from
placeu.lvp_placeu;
quit;

/*Merge Cleaned Consultant Data*/
data placeu.all_consultants;
set placeu.stagearea placeu.stageareaMan placeu.stageareaLiv;
run;

/*Create Consultant Sequence & Remove Missing Values*/
data placeu.all_consultants;
Set placeu.all_consultants;
consultant_id = _N ;
if (cst_end = .) then cst_end = today();
run;

/*Insert Into Consultant Dim*/
proc sql;
insert into placeu.dim_consultant select consultant_id, cst_name,
cst_start, cst_end from placeu.all_consultants;
quit;

```

```

/*Location Staging Area*/
proc sql;
Create table placeu.placeu_location1 as select DISTINCT location_id,
location_name, location_post from placeu.lds_placeu;
Create table placeu.placeu_location2 as select DISTINCT location_id,
location_name, location_post from placeu.mch_placeu;
Create table placeu.placeu_location3 as select DISTINCT location_id,
location_name, location_post from placeu.lvp_placeu;
quit;

/*Merge Cleaned Location Data*/
data placeu.placeu_locations;
set placeu.placeu_location1 placeu.placeu_location2
placeu.placeu_location3;
run;

/*Insert Into Location Dim*/
data placeu.dim_location ;
set placeu.placeu_locations ;
keep location_id location_name location_post;
run;

/*Re-Merge Cleaned Consultant & Location Data*/
proc sql;
Create table placeu.merge1 as select consultant_id, location_id,
cst_name, cst_start, cst_end from placeu.lds_placeu;
Create table placeu.merge2 as select consultant_id, location_id,
catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from
placeu.mch_placeu;
create table placeu.merge3 as select consultant_id, location_id,
catx(' ',cst_fname, cst_sname) AS cst_name, cst_start, cst_end from
placeu.lvp_placeu;
quit;

data placeu.all_consultants_remerge;
set placeu.merge1 placeu.merge2 placeu.merge3;
run;

data placeu.all_consultants_remerge;
Set placeu.all_consultants_remerge;
consultant_id = _N_;
if (cst_end = .) then cst_end = today();
run;

/*Reward Fact Staging Area*/
proc sql;
CREATE TABLE placeu.fact_stage
(
    consultant_id INTEGER,
    location_id INTEGER,
    time_id INTEGER,
    cst_start DATE,
    cst_start_year INTEGER,
    cst_end DATE,
    days INTEGER,
    months INTEGER,

```



```

        years INTEGER
    );

insert into placeu.fact_stage select dc.consultant_id,
ld.location_id, dt.time_id, dc.cst_start, year(dc.cst_start) AS
cst_start_year,
dc.cst_end, datdif(dc.cst_start, dc.cst_end, 'act/act')AS days,
INTCK('MONTH', dc.cst_start, dc.cst_end, 'C') AS months,
INTCK('YEAR', dc.cst_start, dc.cst_end, 'C') AS years
from placeu.dim_consultant dc, placeu.all_consultants_remerge acr,
placeu.dim_time dt, placeu.dim_location ld
WHERE dc.consultant_id = acr.consultant_id
AND ld.location_id = acr.location_id
AND dt.year = year(dc.cst_start);
quit;

/*Reward Fact Temp Table*/
proc sql;
CREATE TABLE placeu.fact_temp
(
    consultant_id INTEGER,
    location_id INTEGER,
    time_id INTEGER,
    days_of_service INTEGER,
    months_of_service INTEGER,
    years_of_service INTEGER
);

insert into placeu.fact_temp select consultant_id, location_id,
time_id, days AS days_of_service, months AS months_of_service, years
AS years_of_service
from placeu.fact_stage
group by consultant_id;

quit;

/*Create Reward Fact Sequence & Load Into Reward Fact*/
data placeu.FACT_reward;
Set placeu.fact_temp;
reward_id = _N_;
run;

/*Report 1 to look at the number of consultants that have worked for
the company
for 5+ years in the Leeds office in comparison to the Manchester and
Liverpool offices*/

PROC SQL;
title 'Consultants 5+ Years';
select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL =
"Consultant Name",
ld.location_name LABEL = "Office Location", rf.years_of_service
LABEL = "Years of Service"
from placeu.FACT_reward rf, placeu.dim_consultant cd,
placeu.dim_location ld
where rf.consultant_id = cd.consultant_id

```

```

and rf.location_id = ld.location_id
and rf.years_of_service >= 5
order by rf.years_of_service;
quit ;

/* Set Graphics Environment */
goptions reset=all border cback=white htext=10pt htitle=12pt;

/* Create Data Set for Graph */

proc sql;
CREATE TABLE placeu.consultant_graph
(
    cst_name varchar (255),
    years_of_service INTEGER
);

insert into placeu.consultant_graph select c.cst_name,
r.years_of_service
from placeu.FACT_reward r, placeu.dim_consultant c
where r.consultant_id = c.consultant_id;

quit;

/*Titles and Labels*/
axis1 label=('Consultants');
axis2 label=('Years Of Service');
title1 'Consultants Years of Service';

/* Create Graph */
proc gchart data=placeu.consultant_graph;
hbar cst_name/ maxis=axis1 raxis=axis2 nostats
sumvar=years_of_service;
run;

/*Report 2 to look at the number of consultants that have started
working for the company over the last year*/
PROC SQL;
title 'Consultants New Starters';
select rf.consultant_id LABEL = "Consultant ID", cd.cst_name LABEL =
"Consultant Name",
ld.location_name LABEL = "Office Location", cd.cst_start LABEL =
"Start Date"
from placeu.FACT_reward rf, placeu.dim_consultant cd,
placeu.dim_location ld
where rf.consultant_id = cd.consultant_id
and rf.location_id = ld.location_id
and cd.cst_start between "01jan15"d and today()
order by rf.consultant_id;
quit ;

/* Set Graphics Environment */
goptions reset=all border cback=white
        htitle=12pt htext=10pt;

```

```

/* Create Data Set for Graph */

proc sql;
CREATE TABLE placeu.consultant_plot
(
    consultant_id INTEGER,
    years_of_service INTEGER,
    location_name VARCHAR (255)
);

insert into placeu.consultant_plot select c.consultant_id,
r.years_of_service, l.location_name
from placeu.FACT_reward r, placeu.dim_consultant c,
placeu.dim_location l
where r.consultant_id = c.consultant_id
and r.location_id = l.location_id;

quit;

/*Titles and Labels*/
title1 'Consultants Years of Service by Location';
axis1 order = (1 to 33 by 1);
axis2 order=(1 to 16 by 1);
symbol1 interpol=join value=dot height=1.7;

/* Create Graph */
proc gplot data= placeu.consultant_plot;
plot years_of_service*consultant_id=location_name / hminor=9;
run;

```