Database Design and Implementation
HW 03

Team 08
Henriod, Terence
Santoyo, Jorge
Singh, Raja

February 9, 2015

Abstract

An introductory assignment to become familiarized with creating and populating database tables using SQL.

1 Assignment Background

The purpose of this assignment is to create tables with constraints and then populate those tables with data to produce a test database for CutGlass Mosaic & Tile, a hard surface (tile, glass, marble, granite, etc.) subcontracting firm. CutGlass provides hard surface removal, design, installation, and maintenance services. The company creates complex, artistic mosaics as well as simple, standard tile floors. CutGlass employs a range of workers including artistic designers, surface removal specialists, and tiling craftsmen.

The application for this assignment is part of their job costing system. The purpose of the database is to collect actual labor and material costs and assign those costs to a specific task on a specific job. The job costing system is used to compare actual direct costs for labor and materials to estimated costs for labor and materials by task on a job. In addition, this system will be used to compare actual labor hours to estimated labor hours. Thus, employee hours are also assigned to a task on a job.

Your objective for this assignment is to create ten tables in your existing database as described in this document and populate those tables with the data included on the pages following the table descriptions. Please do not make up different data – you must use the data included in this document. I want everyone in class to be working from exactly the same test dataset.

This assignment is not especially difficult, but it may be time-consuming. It is a good way to become familiar with SQL syntax and Microsoft's SQL Server, so bear with it - the next three assignments build on this one and are more challenging and fun!

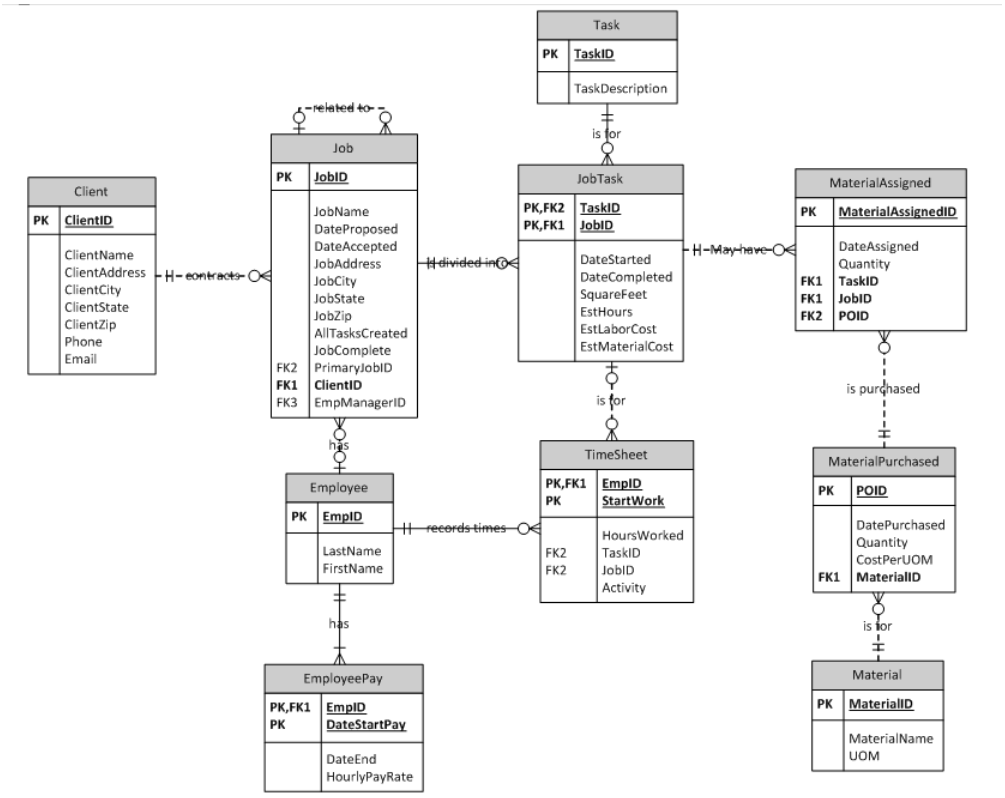

Figure 1: The solution ERD for Chemical Engineering Company's database.

2 Creating Tables

Give the `CREATE TABLE` statements used to create each of the tables.

Solution:

```
CREATE TABLE Client(
    ClientID int NOT NULL,
    ClientName varchar(30) NOT NULL,
    ClientAddress varchar(30) NOT NULL,
    ClientCity varchar(15) NOT NULL,
    ClientState char(32) NOT NULL,
    ClientZip varchar(12) NOT NULL,
    Phone char(10) NOT NULL,
    Email varchar(50),
    CONSTRAINT pk_ClientID PRIMARY KEY(ClientID)
);

CREATE TABLE Employee(
    EmpID int NOT NULL,
    LastName varchar(30) NOT NULL,
    FirstName varchar(30),
    CONSTRAINT pk_EmpID PRIMARY KEY(EmpID)
);

CREATE TABLE Task(
    TaskID int NOT NULL,
    TaskDescription varchar(50) NOT NULL,
    CONSTRAINT pk_TaskID PRIMARY KEY(TaskID)
);

CREATE TABLE Job(
    JobID int NOT NULL,
    JobName varchar(40) NOT NULL,
    DateProposed Date NOT NULL,
    DateAccepted Date,
    JobAddress varchar(30),
    JobCity varchar(15),
    JobState char(2),
    JobZip varchar(12),
    AllTaskCreated Bit NOT NULL,
    JobCompleted Bit NOT NULL,
    PrimaryJobID int,
    ClientID int NOT NULL,
    EmpManagerID int,
    CONSTRAINT pk_JobID PRIMARY KEY(JobID),
    CONSTRAINT fk_PrimaryJobID FOREIGN KEY (PrimaryJobID) REFERENCES Job(JobID),
    CONSTRAINT fk_ClientID FOREIGN KEY (ClientID) REFERENCES Client(ClientID),
    CONSTRAINT fk_EmpManagerID FOREIGN KEY (EmpManagerID) REFERENCES Employee(EmpID)
);

CREATE TABLE JobTask(
    TaskID int NOT NULL,
    JobID int NOT NULL,
    DateStarted Date NOT NULL,
```

```
    DateCompleted Date,
    SquareFeet int NOT NULL,
    EstHours int NOT NULL,
    EstLaborCost Money NOT NULL,
    EstMaterialCost Money NOT NULL,
    CONSTRAINT pk_JobTask PRIMARY KEY(TaskID,JobID),
    CONSTRAINT fk_TaskID FOREIGN KEY(TaskID) REFERENCES Task(TaskID),
    CONSTRAINT fk_JobID FOREIGN KEY(JobID) REFERENCES Job(JobID)
);

CREATE TABLE EmployeePay(
    EmpID int NOT NULL,
    DateStartPay Date NOT NULL,
    DateEnd Date,
    HourlyPayRate Money NOT NULL,
    CONSTRAINT pk_employeePay PRIMARY KEY (EmpID,DateStartPay),
    CONSTRAINT fk_EmpID FOREIGN KEY (EmpID) REFERENCES Employee(EmpID)
);

CREATE TABLE TimeSheet(
    EmpID int NOT NULL,
    StartWork DateTime NOT NULL,
    HoursWorked Decimal (4,2) NOT NULL,
    TaskID int,
    JobID int,
    Activity varchar(15),
    CONSTRAINT pk_TImeSheet PRIMARY KEY(EmpID,StartWork),
    CONSTRAINT fk_TimeSheetEmpID FOREIGN KEY(EmpID) REFERENCES Employee(EmpID),
    CONSTRAINT fk_JobTaskID FOREIGN KEY(TaskID,JobID) REFERENCES JobTask(TaskID,JobID),
);


CREATE TABLE Material(
    MaterialID int NOT NULL,
    MaterialName varchar(50) NOt NULL,
    UOM varchar(5),
    CONSTRAINT pk_MaterialID PRIMARY KEY(MaterialID),
    CHECK ( UOM = 'Tube' OR  UOM = 'SQFT' OR UOM = 'Sheet' OR
            UOM = 'qt' OR UOM = 'lb' OR UOM = 'gal' OR UOM = 'ea' OR  UOM = 'ft' OR
            UOM ='ton' OR UOM  = 'pint' )
);

CREATE TABLE MaterialPurchased(
    POID int NOT NULL,
    DatePurchased DateTime NOT NULL,
    Quantity Decimal (7,2) NOT NULL,
    CostPerUOM Money NOT NULL,
    MaterialID int NOT NULL,
    CONSTRAINT pk_POID PRIMARY KEY(POID),
    CONSTRAINT fk_MaterialID FOREIGN KEY(MaterialID) REFERENCES Material(MaterialID)
);

CREATE TABLE MaterialAssigned(
    MaterialAssignedID int NOT NULL IDENTITY (1000,1),
```

```
    DateAssigned DateTime NOT NULL,
    Quantity Decimal(7,2) NOT NULL,
    TaskID int NOT NULL,
    JobID int NOT NULL,
    POID int NOT NULL,
    CONSTRAINT pk_MaterialAssigned PRIMARY KEY(MaterialAssignedID),
    CONSTRAINT fk_MaterialAssignedJobTaskID FOREIGN KEY(TaskID,JobID) REFERENCES JobTask(TaskID,Job
    CONSTRAINT fk_POID FOREIGN KEY(POID) REFERENCES MaterialPurchased(POID)
);
```

3 Table Population Methods

Describe how you populated the tables (copied and pasted into Excel, used a program to insert `INSERT` statements, etc.) but do not submit the actual SQL code used.

Solution:
In order to populate the tables, data was copied and pasted from the assignment specification's Word document into Excel. Then Python scripts that required slight customization for each case were used to parse and reformat the data values into SQL code that would insert the data into the tables. The Python scripts handled the parenthesizing of data tuples, added or removed commas as appropriate, added commas and semicolons as appropriate, and added or removed quote marks as appropriate for the various data types. The results were then copied and pasted into a main SQL query file for easy re-use.

4 Table Contents

Include the contents of each table as given by a query of the form
`SELECT * FROM TableName`.

Solution:
The output of the `Client` table:

ClientID	ClientName	ClientAddress	ClientCity	ClientState	ClientZip	Phon
2417	Kelly Property Development	66750 Industrial Parke Drive	Sparks	NV	89431	77586
4339	3 Gals From Verona	1001 Apple Road	Sparks	NV	89431	77542
4469	Fran and Harrold Meyers	5740 Braeburn Hill Drive	Reno	NV	89509	77586
5012	Less Furniture Company	1200 Apricot Blvd.	Reno	NV	89509	77570
6295	AO Reid Construction	4275 Contractor Terrace	Sparks	NV	89431	77572
6672	Ms. Catherine Hampstead	23 Pine Forest Ave.	Incline Village	NV	89541	77591
9948	Adam's Rib Restaurant	27 Main Street	Truckee	CA	96161	53090

The output of the `Client` table:

ClientID	ClientName	ClientAddress	ClientCity	ClientState	ClientZip	Phon
2417	Kelly Property Development	66750 Industrial Parke Drive	Sparks	NV	89431	77586
4339	3 Gals From Verona	1001 Apple Road	Sparks	NV	89431	77542
4469	Fran and Harrold Meyers	5740 Braeburn Hill Drive	Reno	NV	89509	77586
5012	Less Furniture Company	1200 Apricot Blvd.	Reno	NV	89509	77570
6295	AO Reid Construction	4275 Contractor Terrace	Sparks	NV	89431	77572
6672	Ms. Catherine Hampstead	23 Pine Forest Ave.	Incline Village	NV	89541	77591
9948	Adam's Rib Restaurant	27 Main Street	Truckee	CA	96161	53090

The output of the `Employee` table:

EmpID	LastName	FirstName
2300	Riggs	Evelyn
2480	Hazelton	Blake
3155	Allen	Noel
4702	Walker	Vance Martin
5291	Kinney	Deneece
5862	Bridges	Carol
6460	Kane	Sylvia
7651	Galloway	Odessa
7656	Wiggins	Cody
8110	Burgess	DaraLee
8750	Fleming	Cathleen

The output of the `EmployeePay` table:

EmpID	DateStartPay	DateEnd	HourlyPayRate
2300	1/9/2012	NULL	12
2480	5/2/2011	5/1/2012	13.5
2480	5/2/2012	NULL	14
3155	3/7/2012	3/6/2013	9.75
3155	3/7/2013	3/6/2014	10.25
3155	3/7/2014	NULL	10.75
4702	3/24/2011	NULL	23
5291	3/9/2011	3/8/2012	15.5
5291	3/9/2012	NULL	16
5862	8/15/2012	11/4/2014	16
5862	11/5/2014	NULL	16.72
6460	11/8/2011	2/3/2014	24.5
6460	2/4/2014	NULL	27.5
7651	11/1/2012	NULL	21
7656	11/11/2011	12/18/2014	11.5
7656	12/19/2014	NULL	12.15
8110	10/1/2012	1/31/2014	10
8110	2/1/2014	NULL	11
8750	1/2/2013	NULL	14

The output of the `Job` table:

JobID	JobName	DateProposed	DateAccepted	JobAddress	JobCity	JobSta
78431	Custom Stained Glass Part.	11/1/2014	11/18/2014	1500 Diagon Alley	Reno	NV
91584	Restaurant Remodel	3/18/2014	3/20/2014	27 Main Street (Outside Bar)	Truckee	CA

The output of the `JobTask` table:

TaskID	JobID	DateStarted	DateCompleted	SquareFeet	EstHours	EstLaborCost	EstMaterialCost
110	16885	12/10/2014	12/11/2014	95	6	75	25
110	32687	7/31/2014	7/31/2014	100	8	88	0
130	16885	12/12/2014	12/16/2014	160	22	280	360
130	32687	8/4/2014	8/4/2014	100	7	113	95
130	55841	7/15/2013	7/15/2013	94	8	83	83
130	55873	8/14/2013	8/15/2013	94	8	110	83
130	55878	9/12/2013	9/12/2013	94	8	110	83
140	16885	11/24/2014	11/28/2014	32	10	240	10
140	78431	11/17/2014	12/15/2014	160	24	575	50
150	16885	12/16/2014	12/16/2014	160	5	61	250
150	32687	8/4/2014	8/7/2014	100	3	50	110
150	55841	7/16/2013	7/16/2013	94	4	45	110
150	55873	8/16/2013	8/16/2013	94	4	45	110
150	55878	9/16/2013	9/16/2013	94	4	45	110
150	62254	10/14/2014	10/24/2014	990	20	270	790
150	62257	10/30/2014	11/3/2014	990	20	270	790
150	91584	3/31/2014	4/2/2014	1240	33	450	300
160	16885	12/17/2014	NULL	160	18	230	1530
160	32687	8/11/2014	8/14/2014	100	10	180	13550
160	55841	7/18/2013	7/19/2013	94	11	170	240
160	55873	8/19/2013	8/20/2013	94	11	170	240
160	55878	9/19/2013	9/20/2013	94	11	170	240
160	62254	10/16/2014	10/30/2014	990	65	1040	1897
160	62257	11/3/2014	11/7/2014	990	65	1040	1897
160	91584	4/7/2014	4/11/2014	1274	72	1200	11590
170	16885	12/28/2014	NULL	300	33	450	40
170	32687	8/18/2014	8/18/2014	100	6	90	12
170	55841	7/22/2013	7/22/2013	94	10	140	9
170	55873	8/26/2013	8/26/2013	94	10	140	9
170	55878	9/24/2013	9/24/2013	94	10	140	9
170	62254	10/21/2014	10/31/2014	990	40	540	120
170	62257	11/7/2014	11/12/2014	990	40	540	120
170	91584	4/14/2014	4/14/2014	1274	16	216	150
180	16885	1/19/2015	NULL	160	17	180	25
180	32687	8/22/2014	8/22/2014	100	4	60	10
180	55841	7/29/2013	7/29/2013	94	5	58	13
180	55873	9/2/2013	9/2/2013	94	5	58	13
180	55878	9/30/2013	9/30/2013	94	5	53	13
180	62254	10/30/2014	11/3/2014	990	33	446	125
180	62257	11/13/2014	11/18/2014	990	33	446	125
180	91584	4/18/2014	4/21/2014	1274	8	108	175
190	78431	1/19/2015	NULL	160	4	64	0
200	78431	1/12/2015	NULL	160	8	128	175
230	91584	4/3/2014	4/4/2014	34	16	265	400
240	16885	12/22/2014	NULL	140	16	252	1625
260	16885	1/5/2015	NULL	32	16	384	110
260	78431	12/22/2014	NULL	160	40	960	1460
270	16885	12/8/2014	12/10/2014	112	8	100	0

The output of the `Material` table:

MaterialID	MaterialName	UOM
1010	Ard D 14 Type 1 Tile Adhesive	GAL
1020	Ard AF 207 Rapid Set Bonding	GAL
1030	Henry 440 Bulk	GAL
1040	Henry 440 Cove Base	TUBE
1050	Henry 356C Multi	GAL
1060	Chapco Moisture Defender	GAL
1070	Ceramic Tile	SQFT
1080	Porcelain Tile	SQFT
1090	Mosaic/Stained Glass	SQFT
1100	Granite	SQFT
1110	Hardie Backer	SHEET
1120	Underlayment Screws	SHEET
1130	Standard Install Supply Pack	EA
1140	ProTex Concrete Backer	SQFT
1170	Tile and Stone sealant	QT
2000	Misc Material	EA

The output of the `MaterialsAssigned` table:

MaterialAssignedID	DateAssigned	Quantity	TaskID	JobID	POID		
1000	7/15/2013	0.5	130	55841	75697		
1001	7/15/2013	7	130	55841	63241		
1002	7/15/2013	15	130	55841	35874		
1003	8/13/2013	0.5	130	55873	75697		
1004	8/13/2013	7	130	55873	63241		
1005	8/13/2013	15	130	55873	35874		
1006	9/12/2013	0.5	130	55878	75697		
1007	9/12/2013	6	130	55878	63241		
1008	9/12/2013	12	130	55878	35874		
1009	7/16/2013	0.5	150	55841	83268		
1010	8/16/2013	0.5	150	55873	83268		
1011	9/16/2013	0.5	150	55878	83268		
1012	3/31/2014	313.87	150	91584	44342		
1013	4/1/2014	1300	160	91584	35182		
1014	4/3/2014	45	230	91584	90090		
1015	7/18/2013	0.5	160	55841	58707		
1016	7/18/2013	110	160	55841	92573		
1017	8/19/2013	0.5	160	55873	58707		
1018	8/19/2013	2	160	55873	92573		
1019	9/19/2013	0.5	160	55878	58707		
1020	9/19/2013	3	160	55878	92573		
1021	7/22/2013	12	170	55841	36751		
1022	8/26/2013	12	170	55873	36751		
1023	7/22/2013	12	170	55878	36751		
1024	7/29/2013	0.5	180	55841	58836		
1025	9/2/2013	0.5	180	55873	58836		
1026	9/30/2014	0.5	180	55878	58836		
1027	4/1/2014	7	160	91584	71055		
1028	4/14/2014	163	170	91584	67046		
1029	4/18/2014	7	180	91584	50075		
1030	8/4/2014	8	130	32687	41911		
1031	8/4/2014	1	130	32687	68203		
1032	8/8/2014	0.5	150	32687	60169		
1033	8/11/2014	1	160	32687	24325		
1034	8/11/2014	120	160	32687	65120		
1035	8/18/2014	13	170	32687	23769		
1036	8/22/2014	0.5	180	32687	63572		
1037	10/14/2014	5	150	62254	81478		
1038	10/14/2014	70	150	62254	40800		
1039	10/30/2014	5	150	62257	81478		
1040	10/30/2014	70	150	62257	40800		
1041	10/16/2014	50	160	62254	91343		
1042	10/16/2014	5	160	62254	82408		
1043	10/21/2014	125	170	62254	10475		
1044	10/29/2014	5	180	62254	63572		
1045	11/3/2014	50	160	62257	91343		
1046	11/3/2014	5	160	62257	82408		
1047	11/8/2014	125	170	62257	48035		
1048	11/13/2014	5	180	62257	63572		
1049	12/9/2014	3	140	16885	35874		
1050	12/9/2014	15	140	16885	94515		
1051	12/14/2014	0	130	16885	40800		
1052	12/15/2014	8	130	16885	95998		
1053	12/15/2013	157	130	16885	94515		
1054	12/14/2014	0.5	130	16885	75697		
1055	12/16/2014	0.5	150	16885	60169		
1056	12/16/2014	0.5	150	16885	83268		
1057	12/16/2014	38	150	16885	94515		
1058	12/17/2014	0.5	160	16885	58707		

The output of the `MaterialsPurchased` table:

POID	DatePurchased	Quantity	CostPerUOM	MaterialID
10475	10/21/2014	125	0.93	1150
23769	8/18/2014	13	0.99	1150
24325	8/4/2014	1	10.6	1010
35182	4/1/2014	1300	8.95	1080
35874	7/10/2013	45	1	2000
36751	4/1/2014	40	0.99	1150
40800	10/6/2014	150	10.25	1110
41911	8/4/2014	8	10.75	1110
44342	3/17/2013	313	1	2000
48035	11/6/2014	125	0.93	1150
50075	4/1/2014	7	24	1170
57713	12/11/2014	160	12.14	1080
58707	7/9/2013	2	10.5	1010
58836	7/15/2013	2	23.5	1170
60169	8/2/2014	1	210	1060
62263	12/5/2014	180	8.15	1090
63241	7/9/2013	20	10.5	1140
63572	8/20/2014	15	19.95	1170
65120	8/4/2014	120	115	1080
67046	4/1/2014	163	0.95	1150
68203	8/4/2014	1	12.95	1020
71055	4/1/2014	7	13	1010
75697	7/9/2013	2	12.8	1020
81478	10/6/2014	12	10.75	1020
82408	10/6/2014	12	11.09	1010
83268	7/9/2013	2	199.95	1060
90090	4/1/2014	45	9.55	1080
91343	10/7/2014	2100	1.75	1070
92573	7/15/2013	320	2.09	1070
94515	1/1/2014	500	1	2000
94696	12/29/2014	48	10.78	1090
95998	12/12/2014	8	12.25	1110
96900	12/22/2014	40	1.15	1150
97277	12/11/2014	180	9.88	1080

The output of the `Task` table:

TaskID	TaskDescription
110	Remove Existing Floor
120	Clean mold
130	Install sub-floor
140	Design mosaic/glass
150	Prepare sub floor
160	Install tile floor
170	Grout
180	Seal and finish work
190	Install mosaic
200	Build structure
210	Remove existing tile
220	Clean site
230	Install tile counter
240	Install tile wall
260	Build mosaic/glass
270	General demolition

The output of the `TimeSheet` table:

EmpID	StartWork	HoursWorked	TaskID	JobID	Activity
2300	7/15/13 8:00 AM	3.5	130	55841	NULL
2300	7/15/13 1:30 PM	4	130	55841	NULL
2300	7/16/13 8:30 AM	4	150	55841	NULL
2300	7/22/13 8:00 AM	4	170	55841	NULL
2300	9/12/13 8:00 AM	4	130	55878	NULL
2300	9/13/13 1:00 PM	4.5	150	55878	NULL
2300	9/25/13 7:30 AM	4	170	55878	NULL
2300	9/29/13 12:00 PM	5	180	55878	NULL
2300	4/14/14 8:30 AM	3.75	170	91584	NULL
2300	4/14/14 2:00 PM	2	170	91584	NULL
2300	4/15/14 9:00 AM	3	170	91584	NULL
2300	4/15/14 1:00 PM	2.5	170	91584	NULL
2300	4/22/14 12:45 PM	3.75	180	91584	NULL
2300	8/8/14 1:00 PM	4.5	150	32687	NULL
2300	8/15/14 8:00 AM	4	170	32687	NULL
2300	8/15/14 1:00 PM	2.5	170	32687	NULL
2300	10/24/14 8:00 AM	4	170	62254	NULL
2300	10/24/14 1:00 PM	4	170	62254	NULL
2300	10/27/14 8:00 AM	3	170	62254	NULL
2300	10/27/14 1:00 PM	2.5	170	62254	NULL
2300	12/15/14 8:00 AM	4	130	16885	NULL
2300	12/15/14 1:00 PM	4	130	16885	NULL
2300	12/16/14 8:15 AM	3.75	130	16885	NULL
2300	12/16/14 12:30 PM	4.5	150	16885	NULL
2480	7/22/13 8:00 AM	4	170	55841	NULL
2480	7/22/13 1:00 PM	2.5	170	55841	NULL
2480	8/16/13 9:00 AM	3	150	55873	NULL
2480	8/16/13 12:00 PM	1	NULL	NULL	Lunch
2480	8/16/13 1:00 PM	1.25	150	55873	NULL
2480	8/26/13 7:30 AM	4	170	55873	NULL
2480	8/26/13 2:00 PM	3	170	55873	NULL
2480	9/2/13 8:15 AM	5	180	55873	NULL
2480	9/12/13 8:00 AM	4	130	55878	NULL
2480	9/25/13 8:00 AM	3.5	170	55878	NULL
2480	9/25/13 1:00 PM	1	NULL	NULL	Med Appointment
2480	9/25/13 2:00 PM	2.25	170	55878	NULL
2480	3/31/14 8:30 AM	3.5	150	91584	NULL
2480	3/31/14 1:00 PM	4	150	91584	NULL
2480	4/1/14 8:00 AM	4	150	91584	NULL
2480	4/1/14 1:00 PM	4	150	91584	NULL
2480	4/2/14 8:30 AM	3	150	91584	NULL
2480	8/21/14 8:30 AM	4.5	180	32687	NULL
2480	10/14/14 9:00 AM	3.5	150	62254	NULL
2480	10/14/14 12:30 PM	0.3	NULL	NULL	lunch
2480	10/14/14 1:00 PM	4	150	62254	NULL
2480	10/15/14 8:00 AM	4	150	62254	NULL
2480	10/15/14 1:00 PM	4	150	62254	NULL
2480	10/16/14 8:00 AM	4	150	62254	NULL
2480	10/17/14 12:00 PM	1	NULL	NULL	Lunch
2480	11/10/14 1:00 PM	4	170	62257	NULL
2480	11/11/14 8:00 AM	4	170	62257	NULL
2480	11/12/14 8:00 AM	2.5	170	62257	NULL
2480	11/12/14 1:00 PM	1	170	62257	NULL
3155	8/26/13 8:00 AM	3.5	170	55873	NULL
3155	3/31/14 8:30 AM	3.5	150	91584	NULL
3155	3/31/14 1:00 PM	4	150	91584	NULL
3155	4/1/14 8:00 AM	4	150	91584	NULL
3155	4/1/14 1:00 PM	4	150	91584	NULL
3155	10/30/14 12:30 PM	4.25	180	62254	NULL