Section #1

1.

```
proc import datafile="drugclaims.csv" out=e2 dbms=csv;
run;

proc sort data=e2; by bene_id; run;
data e3; set e2;
by bene_id;
if first.bene_id then cum_sum=0;
cum_sum+drug_cost;
run;

proc print data=e3; run;

proc sql;
create table e4 as
select bene_id,claim_dt
from e3 a
where cum_sum=  (select min(cum_sum) from  e3
where bene_id=a. bene_id and cum_sum>=275)
;
select * from e4;
quit;
```

2.

a.

a.	5/1/1941
b.	6/1/1939
c.	3/1/1942
d.	8/1/1952
e.	1/1/1941

(see b. for reasoning)

b.
A sequential algorithm to validate DOB is suggested as below.

1. Data consistency check (If calculated age Year-DOB < 65 and CREC=1 discard DOB.)
2. Value range check (Data in abnormal range will be discarded.)
3. If there are multiple DOBs, choose most frequent ones. Do random selection if there are multiple qualifying records.

Exception 1: if there is the first claim record due to age eligibility, choose DOB in that year.
Exception 2: if there is no valid DOB but a record with age eligibility, impute DOB by calculating DOB in that year (i.e. (Year-65)/01/01). If there are multiple qualifying years, choose the earliest.

```
It can be implemented as below.

proc import datafile="birth_dates.csv" out=t dbms=csv replace ;
run;

proc sql;
create table t as
select *,-intck('year','01jan2006'd,dob1,'c') as age1,
 -intck('year','01jan2007'd,dob2,'c') as age2,
 -intck('year','01dec2008'd,dob3,'c') as age3
from t;
/*1. impute 1: range check */
create table t as
select *,case when age1>110 then . else dob1 end as dob_imputed1_
,case when age2>110 then . else dob2 end as dob_imputed2_
,case when age3>110 then . else dob3 end as dob_imputed3_
from t;
/*2. impute 2: data consistency check */
create table t a
select *,
case when age1<65 and crec1=1 then . else dob_imputed1_  end as dob_imputed1
,case when age2<65 and crec2=1 then . else dob_imputed2_  end as dob_imputed2
,case when age3<65 and crec3=1 then . else dob_imputed3_  end as dob_imputed3
from t;
select * from t;
quit;

/* impute 3: if there is the first claim record due to age eligibility,
choose DOB from that year */
data test;
set t;
array a {*} age1-age3;
array b {*} crec1-crec3;
array d {*} cl1-cl3;
array c {*} dob1-dob3;
dob_first=.;
do i=1 to dim(a);
 if dob_first=. then do;
     if a[i]=65 and b[i]=1 and d[i]="Y" then dob_first=c[i];
  end;
 end;
run;
data test;
set test;
array c dob1-dob3;
if dob_first^=. then
do over c;
c=dob_first;
end;
run;

/* impute 4: if no valid DOB, impute DOB with assuming min qualifying age at
the first eligible year */
data test2;
set test;
array a {*} dob_imputed1-dob_imputed3;
array b {*} crec1-crec3;
array c [3] (2006 2007 2008);
x=sum(of a{*});
dob_avail_first=.;
if x=. then do;
do i=1 to dim(a);
   if dob_avail_first=. then do;
     if b[i]=1 then dob_avail_first=mdy(1,1,c[i]-65);;
  end;
 end;
end;
run;

data test2;
set test2;
array c dob1-dob3;
if dob_avail_first^=. then
do over c;
c=dob_avail_first;
end;
run;

/*after imputation, choose the most frequent DOB */
proc transpose data=test2 (keep=id dob1 dob2 dob3) out=x; by id ; run;

proc sql;
create table c as select id, col1,count(col1) as c from x
group by id,col1;
select  id,  max(col1)
/* random selection if there are still duplicates */
as dob_imputed_final from c as x
where c= (select max(c) from c where c.id=x.id)
group by id;
quit;
```

Section #2

1. R language has several packages for solving a particular problem. How do you make a decision on which one is the best to use?

   There are two packages R basics (loaded by default) and open source extensions, of which the latter the software QA process is up to developers. We often choose most used ones (among ones developed for the same or similar task) since it's more likely that most common issues were already reported and fixed. Also, functionality, documentation, efficient algorithms are some key factors affect the decision.

2. If you want to know all the values in c (1, 3, 5, 7, 10) that are not in c (1, 5, 10, 12, 14). Which builtin function in R can be used to do this? Also, how can this be achieved without using the built-in function?

   1. v1= c (1, 3, 5, 7, 10); v2= c (1, 5, 10, 12, 14); v2[!v2 %in% v1]

2. use ‘sqldf’

```
library(sqldf);
v1=as.data.frame(v1); # repeat for v2
sqldf('select * from v2 except select * from v1 ')
```

3. Write a function in R language to replace the missing value in a vector with the mean of that vector.

```
sapply(x, function(y) if (is.na(y)) mean(x,na.rm=T) else y )
```

4. What are “with()” and “by()” functions used for in R? Please demonstrate by an example.

   1. with( ) function applies an expression to a dataset.
   2. by( ) function applies a function to each level of a factor or factors.

```
#example
data=data.frame(y=c(1,2,3),byvar=c(“A”,”B”))
with(data, aggregate(y ~1,FUN=mean))
by(data$y, data$byvar, function(x) mean (x))
```

5. Write a custom function to get measures of central tendency and spread for a numeric vector x.

```
func1=function(x,choice=1) {
if (choice==1) {
   center <- mean(x); spread <- sd(x)
} else {
center <- median(x); spread <- mad(x) # more accurately describes data with an outlier
}
  result <- list(center=center,spread=spread)
return(result)
}
```

6. Create an R binary decision tree with three node levels. Please list the basic steps and demonstrate by an example.

   1. Generate target with factors as appropriate (so that tree can develop as requested)
   2. Run classification tree (adjust control parameters as appropriate)

```
#example
f1=rnorm(100); f2=rnorm(100,1,1); f3=rnorm(100,2,1);f4=ifelse(f1<0&f2<1&f3<2,1,0)
data=data.frame(y=f4,f1=f1,f2=f2+rnorm(6,0,0.00001)*0,f3=f3+rnorm(6,0,0.00001)*0)
library(rpart)
```

```
rpart(y~.,data, method = "class",control = list(maxdepth = 3,minsplit=1))
```

Section #3

1. What common native data structures are built into Python? Which of them are mutable and which are immutable? Why is this important?

Text Type:	str
Numeric Types:	int, float, complex
Sequence Types:	list, tuple, range
Mapping Type:	dict
Set Types:	set, frozenset
Boolean Type:	bool
Binary Types:	bytes, bytearray, memoryview

Mutable (list, dict, set)
Immutable(int,float,bool,str,tuple, range)

know difference between mutable and immutable if you want to write most efficient code (for example if you want to concat string in loops, you can do it with list comprehension join (string.joing(list). because strings are immutable, concat string in loops wastes lots of memory , because strings are immutable)

2.What is the difference between the “range()” and “xrange()” functions in Python?

Xrange deprecated in python 3. (range in xrange in python 3)
In python 2, range() returns a range object (a iterable). xrange() returns the generator object (Generators allow you to create iterators in a very pythonic manner. Iterators allow lazy evaluation, only generating the next element of an iterable object when requested.)

3.Can you explain what “list comprehensions” mean by an example?

A way to create a list: do in this format
[expression for item in list]

List comprehension is more Pythonic way to write code that is declarative and efficient.

For example instead of using loops

```
 x=[];
for i in range(10):
        x.append(i)
```

you can [i for i in range(10)]

4. What will be the output of the below Python code?

```
def multipliers ():
return [lambda x: i * x for i in range (4)]
print [m (2) for m in multipliers ()]
```

[6,6,6,6] (lambda is called after the loop has finished)

5. Which tool(s) in Python will you use to find bugs, if any? Please demonstrate by an example.

Pychecker is a tool for finding bugs in python source code. Pass options and source files if you want to check on the command line: pychecker [options] file1.py file2.py

6.What packages in the Standard Library are useful for Data Science work? Please explain.

Scikit-Learn: is a Python module for machine learning built on top of SciPy and NumPy (If you’re a beginner and want to pick up a machine learning library, Scikit-Learn is the one to start with)

Pandas : a powerful and flexible data analysis library written in Python. well-suited for data analysis and manipulation for large data sets

NumPy : fundamental package needed for scientific computing with Python

Scipy : builds on top of NumPy, has a wide collection of sub package

Matplotlib: 2d plotting library

Pytorch: an open source machine learning library used for developing and training neural network based deep learning models.

TensorFlow: one of the most highly flexible and powerful deep learning libraries

Keras: a TensorFlow bankend for easy and fast prototyping as a deep learning library.

Section #4

1. Can you describe the elements of an effective data management plan?

Determine sponsor requirements
Determine what data to collect
How data will be organized
How data will be documented
How to ensure quality of data
Determine data storage and preservation, how to disseminate
Define data policy

2. Can you share an effective approach to operating with a large amount of data?

Use storage for backup as you don't want to access the raw data often

Visualize data (Do a lot of graphs and look for outliers)

show data workflow (which version of the data you used, the clean-up and quality-checking steps, and any processing code you ran)

use version control

record metadata

automation (to validate and repair data in real time for ex)

count computing time in (use distributed system for ex)

3. Write a query to get the list of employees who took a training lesson more than once in the same day, grouped by employee and training lesson, each ordered from the most recent lesson date to oldest date.

```
Select tb.emp_id,training_date,training_id, tb1.name from training_details as tb
join employee as tb1
On tb.emp_id= tb1.emp_id
Group by tb.emp_id,training_date,training_id
Having count(*)>1
```

Order by training_date desc

4. What will be the results of the query below, using previous Tables?
SELECT * FROM employee WHERE emp_id NOT IN (SELECT training_id FROM training_details)
Explain your answer and also provide an alternative version of this query that will avoid any issues that it may expose.

Null (both range 1-4)
Why match with training_id ? change training_id to emp_id the subquery.

5. How would you find a duplicate record? Please write a SQL query to find
1) duplicate records with one field
Select * from <table> group by <key> having count(*)>1
2) duplicate records with more than one field
Select * from <table> group by <keys> having count(*)>1
3) to remove duplicates from a table without using temporary table
Try nested query

```
SELECT <keys>,b.<columns>  FROM <table> a

               GROUP BY <keys>  HAVING COUNT(*) = 1

join <table> b on a.<key1>=b.<key1> ………
```