6 Appendix

6.1 Approval from the hospital management to carry out the retrospective study

11.1 ANNEXURE 1 –NOC FROM IAIM MEDICAL DIRECTOR.

****
****

IAIM/2020/NOC/01 Date: 29.05.2020

LETTER OF PERMISSION AND NO OBJECTION CERTIFICATE

TO WHOM IT MAY CONCERN

This is to grant permission to Mr. Vinay Mahajan to conduct the research study "Review of hospital based Ayurvedic Electronic Health Records to gain real world knowledge - a retrospective data analysis" using I-AIM anonymized Electronic Health Records as per the protocol approved by the IEC.

I am assured that Mr. Vinay Mahajan will maintain confidentiality of the data.

Further, it is also agreed that any presentation and publication of the results arising from the study will be done after due permission from the authorities of IAIM and TDU.

Dr. Prasan Shankar
Medical director
IAIM
Bangalore

****
****

6.2 Details of analysis dataset

Table 6-1: Details of the Reference Dataset "01adsl_met_rmsd"

Variable name	Description	Derivation
mr_no	Unique Patient ID	Source variable, no derivation needed E.g. MR000001, MR040237, etc.
patient_gender	Patient gender	Source variable, no derivation needed E.g. M, F
patient_id	Visit ID	Source variable, no derivation needed, the hospital database captures unique visit ID for each visit.
city_name	City name	Source variable, no derivation needed
state_name	State name	Source variable, no derivation needed
country_name	Country name	Source variable, no derivation needed
dateofbirth	Date of birth	Source variable, no derivation needed, for some patients this is missing
newdt0	Date of visit to hospital	Date of visit to hospital in numeric format All the In-Patient visits, Out-Patient visits and Service related visits are combined from source datasets into a dataset, unique visit and date combinations are created.
newdt	Date of visit to hospital	Character version of newdt0

Variable name	Description	Derivation
vis	Visit	1. Based on all the In-Patient visits, Out-Patient visits and Service related visits unique visit numbers are created. 2. Visit numbers are numeric values from 1 to n, based on current version of data; a patient has maximum number 323 visits.
all_vis	All visits	This variable contains maximum number of visit for each patient. all_vis = max(vis) grouped by each mr_no
all_ip	All IP visits	This variable contains maximum number of visits for each patient for IP type of visits. all_vis = max(vis) grouped by each mr_no and visit type is IP.
all_op	All OP visits	This variable contains maximum number of visits for each patient for OP type of visits. all_vis = max(vis) grouped by each mr_no and visit type is OP.
studyday	Study day	studyday = 1 when the visit minimum visit or first visit for a patient, else studyday is calculated as newdt0 – min(newdt0) + 1. Studyday is never missing and never less than 0 for the dataset created.
age	Age of patient at that visit	If date of birth is non-missing for a patient, then age is calculated as round((anydate(newdt) - anydate(dateofbirth) + 1)/365.25, digits = 0)
baseage	Age of patient at the first visit	Age at vis = 1 for each patient is stored as base age

Variable name	Description	Derivation
death_date	Date of death	Source variable, no derivation needed
cstdt	Min Start date	cstdt = min(newdt)
cendt	End date	cendt = max(newdt)
cdur	Total duration in days	cdur = max(newdt) - min(newdt) + 1
stdt_IP	Start date of IP visits	Minimum visit date for IP visits for each patient
endt_IP	End date of IP visits	Maximum visit date for IP visits for each patient
dur_IP	Duration of IP visits	dur_IP = endt_IP – stdt_IP + 1
stdt_OP	Start date of OP visits	Minimum visit date for OP visits for each patient
endt_OP	End date of OP visits	Maximum visit date for OP visits for each patient
dur_OP	Duration of OP visits	dur_OP = endt_OP – stdt_OP + 1
serstdt	Service Start date	Minimum visit date for Service visits for each patient
serendt	Service End date	Maximum visit date for Service visits for each patient
Code	Code	Source variable, no derivation needed, ACD code
description	Description	Source variable, no derivation needed, description
Type	Type of visit	This variable identifies a visit either as IP or OP based on visit classification

Variable name	Description	Derivation
diag_type	Diagnosis type	Source variable, no derivation needed: Primary or Secondary
year	Year	Year part of the newdt variable
season	Indian seasons	Derivation of Indian seasons based on the date variable for each visit: # Add Indian rutus as new variables # https://www.drikpanchang.com/seasons/season-tropical-timings.html?geoname-id=1277333&year=2010 • 01 Vasant Rutu • 02 Grishma Rutu • 03 Varsha Rutu • 04 Sharad Rutu • 05 Hemant Rutu • 06 Shishir Rutu
C, N, P, U, X, Y	Values related to Services offered to patients	Source variable, no derivation needed: • C- Cancelled • U - Condn. Unnecessary • Y -Conducted

Variable name	Description	Derivation
		• N - Not Conducted • P - Partially Conducted
presc_type		Source variable, no derivation needed
medicine_name	Medicine name	Source variable, no derivation needed Prescribed medicine names follow a certain predefined naming convention. Medicine name + Quantity + Producer's name are the details recorded for each prescribed medicine.
item_name	Source value of medicine name	Source variable, no derivation needed
quantity	Quantity of prescribed medicine	Source variable, no derivation needed
med_route	Route of administration of prescribed medicine	Source variable, no derivation needed
generic_code		Source variable, no derivation needed
remarks	Notes provided by doctors for medicines	Source variable, no derivation needed
frequency	Frequency of prescribed medicine	Source variable, no derivation needed
duration	Duration of prescribed medicine	Source variable, no derivation needed

Variable name	Description	Derivation
duration_units	Unit for duration of prescribed medicine	Source variable, no derivation needed
Coded_med	Only name of medicine	Derived from medicine_name
Company	Name of the company producing the drug	Derived from medicine_name
Quantity	Quantity of prescribed medicine	Derived from medicine_name
Unit	Unit of prescribed medicine	Derived from medicine_name
Type_med	Type of medicine	Derived based on medicine_name. Classified into different kinds of medicines, e.g. • Ghritam • Kashayam • Asavam • Aristham • Bhasma • Abhyanga • Cream • Rasayanam

Variable name	Description	Derivation
		• Tablet / Gulika / Vati • …
cat_id		Identification of categories
distype	Disease type	Disease type as OTHER, RMSD, Metabolic 1. If a disease code is present in Metabolic list then the value is Metabolic 2. If a disease code is present in RMSD list then the value is RMSD 3. Any other disease is classified as OTHER
Metabolic	Metabolic	If a patient has reported any Metabolic disease at least once then that patient is given value Metabolic = 1, else Metabolic =0 Metabolic disease group has 10 diseases (Refer 2.4.1.6.1)
RMSD	RMSD	If a patient has reported any RMSD disease at least once then that patient is given value RMSD = 1, else RMSD =0 RMSD disease group has 97 diseases (Refer 2.4.1.6.1)
combine	Metabolic RMSD Both	1. If a patient is classified only as Metabolic diseased patient then combine = 1, 2. If a patient is classified only as RMSD diseased patient then combine = 2, 3. If a patient is classified as Metabolic as well as RMSD diseased patient then combine = 99

Variable name	Description	Derivation
Minday Metabolic	First day on which reported metabolic disease	First day on which any metabolic disease has been reported by a patient.
Minday RMSD	First day on which reported RMSD disease	First day on which any RMSD disease has been reported by a patient.

Table 6-2: Metabolic and RMSD disease code and de-code

Code	Description	Distype
M10.0	Medoroga	Metabolic
M10.1	Medoroga - Sthula medho roga	Metabolic
M10.2	Medoroga - Sukshma medho roga	Metabolic
M2.0	Madhumeha	Metabolic
P5.0	Prameha	Metabolic
P5.1	Prameha - Krusha	Metabolic
P5.2	Prameha - Pidaka	Metabolic
P5.3	Prameha - Sthula	Metabolic
P5.4	Prameha - Upadrava	Metabolic
S16.0	Sthaulya	Metabolic
A2.0	Aamavaata	RMSD
A2.1	Aamavaata - Kaphaja	RMSD
A2.2	Aamavaata - Pittaja	RMSD
A2.3	Aamavaata - Vaataja	RMSD
A3.0	Abhighataja Shoola	RMSD
S10.0	Stambha	RMSD
S10.1	Stambha - Baahu Stambha	RMSD
S10.10	Stambha - Prishtha Stambha	RMSD
S10.12	Stambha - Sandhi Stambha	RMSD
S10.13	Stambha - Siraa Stambha	RMSD
S10.14	Stambha - Uru Stambha	RMSD
S10.4	Stambha - Greevaa Stambha	RMSD
S10.5	Stambha - Hanu Stambha	RMSD
S10.6	Stambha - Hridaya Stambha	RMSD

S13.0	Sthaanabhedena Graha	RMSD
S13.1	Sthaanabhedena Graha - Anga Graha	RMSD
S13.11	Sthaanabhedena Graha - Katee Graha	RMSD
S13.13	Sthaanabhedena Graha - Manyaa Graha	RMSD
S13.14	Sthaanabhedena Graha - Marma Graha	RMSD
S13.17	Sthaanabhedena Graha - Paada Graha	RMSD
S13.18	Sthaanabhedena Graha - Paarshva Graha	RMSD
S13.19	Sthaanabhedena Graha - Prishtha Graha	RMSD
S13.20	Sthaanabhedena Graha - Shiro Graha	RMSD
S13.22	Sthaanabhedena Graha - Uro Graha	RMSD
S13.23	Sthaanabhedena Graha - Vaak Graha	RMSD
S13.3	Sthaanabhedena Graha - Gala Graha	RMSD
S13.5	Sthaanabhedena Graha - Hanu Graha	RMSD
S13.6	Sthaanabhedena Graha - Hrid Graha	RMSD
S13.7	Sthaanabhedena Graha - Jaanugraha	RMSD
S13.8	Sthaanabhedena Graha - Janghaa Graha	RMSD
S14.0	Sthaanabhedena Shoola	RMSD
S14.11	Sthaanabhedena Shoola - Guda Shoola	RMSD
S14.13	Sthaanabhedena Shoola - Gulpha Shoola	RMSD
S14.14	Sthaanabhedena Shoola - Hanu Shoola	RMSD
S14.15	Sthaanabhedena Shoola - Hasta Shoola	RMSD
S14.16	Sthaanabhedena Shoola - Hrid Shoola	RMSD
S14.17	Sthaanabhedena Shoola - Jaanu Shoola	RMSD
S14.18	Sthaanabhedena Shoola - Janghaa Shoola	RMSD
S14.19	Sthaanabhedena Shoola - Kantha Shoola	RMSD
S14.21	Sthaanabhedena Shoola - Katee Shoola	RMSD
S14.23	Sthaanabhedena Shoola - Kukshi Shoola	RMSD

S14.24	Sthaanabhedena Shoola - Manyaa Shoola	RMSD
S14.3	Sthaanabhedena Shoola - Amsa Shoola	RMSD
S14.4	Sthaanabhedena Shoola - Anga Shoola	RMSD
S14.5	Sthaanabhedena Shoola - Anguli Shoola	RMSD
S14.6	Sthaanabhedena Shoola - Asthi Shoola	RMSD
S14.7	Sthaanabhedena Shoola - Baahu Shoola	RMSD
S15.28	Sthaanabhedena Shoola - Nakha Shoola	RMSD
S15.31	Sthaanabhedena Shoola - Paada Shoola	RMSD
S15.32	Sthaanabhedena Shoola - Paarshni Shoola	RMSD
S15.34	Sthaanabhedena Shoola - Parva Shoola	RMSD
S15.36	Sthaanabhedena Shoola - Prishtha Shoola	RMSD
S15.41	Sthaanabhedena Shoola - Sakthi Shoola	RMSD
S15.42	Sthaanabhedena Shoola - Sandhi Shoola	RMSD
S15.43	Sthaanabhedena Shoola - Skandha Shoola	RMSD
S15.44	Sthaanabhedena Shoola - Snaayu Shoola	RMSD
S15.45	Sthaanabhedena Shoola - Sphik Shoola	RMSD
S15.46	Sthaanabhedena Shoola - Stanaanta Shoola	RMSD
S15.47	Sthaanabhedena Shoola - Trika Shoola	RMSD
S15.48	Sthaanabhedena Shoola - Urah Shoola	RMSD
S1A.0	Shoola	RMSD
V1.0	Vaatarakta	RMSD
V1.1	Vaatarakta - Dvandvaja	RMSD
V1.2	Vaatarakta - Gambheera	RMSD
V1.3	Vaatarakta - Kapha Vaataja	RMSD
V1.4	Vaatarakta - Kaphaadhika Vaatarakta	RMSD
V1.5	Vaatarakta - Pittaadhika Vaatarakta	RMSD
V1.7	Vaatarakta - Uttaana	RMSD

V1.8	Vaatarakta - Vaata Kaphaja	RMSD
V1.9	Vaatarakta - Vaataadhika Vaatarakta	RMSD
V2.0	Vaatavyaadhi	RMSD
V2.12	Vaatavyaadhi - Stabdhagaatra	RMSD
V2.16	Vaatavyaadhi - Baahugata Vaata	RMSD
V2.23	Vaatavyaadhi - Gridhrasee	RMSD
V2.30	Vaatavyaadhi - Jaanugata Vaata	RMSD
V2.31	Vaatavyaadhi - Janghaagata Vaata	RMSD
V2.36	Vaatavyaadhi - Kateegata Vaata	RMSD
V2.42	Vaatavyaadhi - Maamsagata Vaata	RMSD
V2.43	Vaatavyaadhi - Maamsamedogata Vaata	RMSD
V2.44	Vaatavyaadhi - Majjaagata Vaata	RMSD
V2.45	Vaatavyaadhi - Majjaasthigata Vaata	RMSD
V2.46	Vaatavyaadhi - Manyaagata Vaata	RMSD
V2.47	Vaatavyaadhi - Manyaastambha	RMSD
V2.48	Vaatavyaadhi - Medogata Vaata	RMSD
V2.61	Vaatavyaadhi - Prishthagata Vaata	RMSD
V2.63	Vaatavyaadhi - Sandhigata Vaata	RMSD
V2.64	Vaatavyaadhi - Sarvaangagata Vaata	RMSD
V2.65	Vaatavyaadhi - Shaakhaagata Vaata	RMSD
V2.68	Vaatavyaadhi - Siraagata Vaata	RMSD
V2.69	Vaatavyaadhi - Siraagraha	RMSD
V2.70	Vaatavyaadhi - Snaayugata Vaata	RMSD
V2.72	Vaatavyaadhi - Trikgata Vaata	RMSD
V2.73	Vaatavyaadhi - Tvaggata Vaata	RMSD
V2.74	Vaatavyaadhi - Urugata Vaata	RMSD
V2.75	Vaatavyaadhi - Vaatakantaka	RMSD

V2.77	Vaatavyaadhi - Vishvaachee	RMSD
V2.9	Vaatavyaadhi - Asthigata Vaata	RMSD

6.3 All variables in the source database

Table 6-3: All variables in the source database

001_all variables in database.xlsx

6.4 Details of analysis

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Figure 3-1: A snippet of disease table by gender	Link	01adsl_met_rmsd	100_adsl.R
Figure 3-2: Variable classification by categories	Link	03_typesOfassessent	03_typesOfassessment.R
Figure 3-3: Visit pattern analysis	Link	01adsl_met_rmsd	100_adsl.R
Figure 3-4: Patient visit profile – Horizontal view	Link	01adsl_met_rmsd	100_adsl.R
Figure 3-5: Patient visit profile – Vertical view	Link	01adsl_met_rmsd	100_adsl.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Figure 3-6: Total Number of Patients	Link	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R
Figure 3-7: Country-wise Visualization	Link	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R
Figure 3-8: Age distribution by country, age distribution by gender	Link01 Link02	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R
Figure 3-9: Blood-group Distribution by gender	Link	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R
Figure 3-10: Number of Visits, and Visit Types	Link	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Figure 3-11: Descriptive summary statistics by number of Diseases by Age and Gender	Link	04_patient_analysis_tableu_adsl	04_patients_analysis_tableu_adsl.R
Figure 3-12: Data tabulation for patients reporting RMSD and Metabolic diseases	Link	rmsd_met_primary_diag	rmsd_metabolic_all.R
Figure 3-13: Disease distribution by age and gender	Link	rmsd_met_primary_diag	rmsd_metabolic_all.R
Figure 3-14: Patient visit duration for	Link	rmsd_met_primary_diag	rmsd_metabolic_all.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Disease categories by Gender			
Figure 3-15: Disease distribution by Seasonal Variations and gender	Link	rmsd_met_primary_diag	rmsd_metabolic_all.R
Figure 3-16: Pre and Post Disease Classification Analysis	Link	085_dis_1st_time_refCal_NodesEdges	085_dis_1st_time_refCal_NodesEdges.R
Figure 3-17: ICD classification by Gender	Link	060_allopathic_diag	060_allopathic_diag.R
Figure 3-18: Age distribution	Link	060_allopathic_diag	060_allopathic_diag.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
by ICD classification and Gender			
Figure 3-19: Visit distribution by ICD classification and Gender	Link	060_allopathic_diag	060_allopathic_diag.R
Figure 3-20: Duration distribution by ICD classification and Gender	Link	060_allopathic_diag	060_allopathic_diag.R
Figure 3-21: Disease classification by Prakriti and Gender	Link	Disease_by_dosha_type	Disease_by_dosha_type.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Figure 3-22: Co-morbidity analysis approach 1 example 1: Vaatavyadhi	Link	prim_diag	diagnosis_primary.R
Figure 3-23: Co-morbidity analysis approach 1 example 2: Pandu	Link	prim_diag	diagnosis_primary.R
Figure 3-24: Co-morbidity analysis approach 1 example 3: Madhumeha	Link	prim_diag	diagnosis_primary.R
Figure 3-25: Co-morbidity	Link	prim_diag_mon	diagnosis_primary_month.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
analysis approach 2			
Figure 3-26: Co-morbidity analysis approach 3: collapsible tree view	Link	085_dis_count_edges_3rd_byPeriod_A2_bruce	085_dis_counts_bruce_java.R
Figure 3-27: Patient Disease and Treatment administration by Study Day	Link	080_disease_repeat_prop	080_medicine_dis_repeat_prop.R
Figure 3-28: Patient Disease by Study Day and Treatment administration by Study Day	Link	080_medicine_dis_all_met_rmsd_prop	080_medicine_dis_repeat_prop.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
Figure 3-29: Patient Cumulative Disease and Treatment administration by Visit	Link	080_medicine_dis_repeat_prop_cumulative	080_medicine_dis_repeat_prop.R
Figure 3-30: Area graph representation of diseases	Link	adiag	diagnosis.R
Figure 3-31: Mosaic plot: Disease and treatment representation example 1: Prameha	Link	305_medicine_duration_by_dis_xyplot	305_medicine_duration_by_dis.R
Figure 3-32: Disease and treatment	This is a part of 3-31 analysis so no other explicit link or program, use the above link		

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
example 2: P5.0: Prameha and Oil: Kottamchukkadi			
Figure 3-33: Disease and treatment example 3: P5.0: Prameha and Vati: Diabecon DS	This is a part of 3-31 analysis so no other explicit link or program, use the above link		
Figure 3-34: Mosaic plot Disease and treatment representation example 4: Treatment: Oil: Kottamchukkadi	Link	305_medicine_duration_by_dis_xyplot	305_medicine_duration_by_dis.R
Figure 3-35: Cross tabulation of prescribed	Link	01adsl_met_rmsd	100_adsl.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
treatments and disease group by gender Example 1			
Figure 3-36: Cross tabulation of prescribed treatments and disease group by gender Example 2	Link	01adsl_met_rmsd	100_adsl.R
Figure 3-37: Circular view: Co-occurrences of disease – disease Example 1	Link	085_dis_counts_edges_3rdbyPeriod_circular17	085_dis_counts_edges_3rdbyPeriod_circular17.R
Figure 3-38: Circular view: Co-occurrences of disease –	Link	085_dis_counts_edges_3rdbyPeriod	085_dis_counts_edges.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
treatment Example 2			
Figure 3-39: Pre and Post distance analysis for disease: M2.0: Madhumeha	Link	086time_dis_refcode_max	086time_dis_patterns_combinations_gender_Macro.R
Figure 3-40: Pre and Post distance analysis for medicines given for diseases: P5.0, V2.23, V2.63	Link	086time_dis_refcode_max	086_med_patterns_combinations.R
Figure 3-41: Radar plot	Link	300_radar_plot	300_radar_plot_tableu.R
Figure 3-42: Dynamic bubble	Link	decode_gender.csv file is converted to Json files	06_d3tree_diagram.R

Figure number and analysis name	Link to visualization	Dataset(s) used, (if needed then the datasets will be made available)	R / SQL / D3js program
plot: Example 1: Disease: A6.0: Amavaata			

6.5 Programs for the different parts of analysis

6.5.1 Data extraction from SQL database: 01adsl.sql

```
SQL program to get the source data from hospital database, combine major components of
data in a logical manner

/* SQL version with UNION of data */

/*=========================================================================*/
/* Execute the code in following manner;                                   */
/* iaim=> \i /cygdrive/d/Hospital_data/ProgresSQL/prgm/100_adsl_sqlpart.sql;  */
/*=========================================================================*/

drop table if exists
temp0pat_demog, temp1pat_reg, temp1doc_cons, temp2reg_cons, temp2diag, temp3pat_presc,
temp4pat_med,
temp20, temp30, temp30_1, temp30_5, temp100ip, temp350, temp100ser, temp100ser2, temp360,
base01_op0, base01_op, base01_ip, base01_ser, base_all ;

/* Country and state names */
create temp table state as
select city.city_id, city.city_name, city.state_id,
state.state_name, state.country_id
from iaim.city as city, iaim.state_master as state
where city.state_id = state.state_id;

create temp table cou as
select state.*, country.country_name
from state, iaim.country_master as country
where state.country_id = country.country_id;

/* Create demog table */
create temp table temp0pat_demog0 as
```

```
select distinct mr_no as mrno, patient_gender, patient_city, patient_state, dateofbirth,
country, /*oldmrno, remarks,*/ death_date
from iaim.patient_details;

create temp table temp0pat_demog as
select cou.city_name, cou.state_name, cou.country_name, temp0pat_demog0.*
from cou, temp0pat_demog0
where cou.city_id = temp0pat_demog0.patient_city and cou.state_id =
temp0pat_demog0.patient_state and cou.country_id = temp0pat_demog0.country;

create temp table temp1pat_reg as
select mr_no, patient_id, visit_type, reg_date, bed_type, dept_name, admitted_dept,
main_visit_id
from iaim.patient_registration
order by mr_no, patient_id;

create temp table temp1doc_cons as
select distinct mr_no as con_mrno, patient_id as con_patient_id, consultation_id as
consult_id, doctor_name, date(visited_date) as visdate
from iaim.doctor_consultation
order by mr_no, patient_id;

create temp table temp2reg_cons as
select temp1pat_reg.*, temp1doc_cons.*
from temp1pat_reg
full join temp1doc_cons on
temp1pat_reg.mr_no = temp1doc_cons.con_mrno and temp1pat_reg.patient_id =
temp1doc_cons.con_patient_id;

/* Create diagnosis table */
create temp table temp2diag as
select distinct visit_id, id, description, icd_code, diag_type, doctor_id,
diagnosis_datetime::timestamptz::date as diagdate
```

```
from iaim.mrd_diagnosis;

/* Full join temp10 and temp2diag on temp10.patient_id and temp2diag.visit_id */
create temp table temp20 as
select temp2reg_cons.*, temp2diag.*
from temp2reg_cons
full join temp2diag on
temp2reg_cons.patient_id = temp2diag.visit_id;

/* patient_prescription = consultation_id */
/* A Subset is required  for presc_type   */
create temp table temp3pat_presc as
select patient_presc_id, consultation_id, presc_type, status, date(prescribed_date) as
dateonly
from iaim.patient_prescription
/*where presc_type in ('Medicine') */
order by patient_presc_id, consultation_id;

create temp table temp30 as
select temp20.*, temp0pat_demog.*
from temp20
full join temp0pat_demog on
temp20.mr_no = temp0pat_demog.mrno;

create temp table temp30_1 as
select temp30.*, temp3pat_presc.*
from temp30
full join temp3pat_presc on
temp30.consult_id = temp3pat_presc.consultation_id;

create temp table temp30_5 as
select mr_no, patient_id, patient_gender, city_name, state_name, dateofbirth,
country_name, death_date,
```

```
consult_id, description, icd_code, diag_type, diagdate, patient_presc_id
from temp30_1;

/* patient_medicine_prescriptions = medicine_id */
create temp table temp4pat_med as
select medicine_id as cat_id,
op_medicine_pres_id,
duration,
duration_units,
mod_time::timestamptz::date as prescdate,
frequency,
medicine_quantity as quantity,
medicine_remarks as remarks
from iaim.patient_medicine_prescriptions
order by medicine_id, op_medicine_pres_id;

/* BASE 1 data for the OP medication */

create temp table base01_op as
select temp30_5.*, temp4pat_med.*
from temp4pat_med
left join temp30_5 on
temp30_5.patient_presc_id = temp4pat_med.op_medicine_pres_id
order by mr_no, patient_id;

/* IP medications */
create temp table temp100ip as
select
prescription_id as consultation_id,
patient_id as ippatient_id,
doctor_id,
prescription_date::timestamptz::date as prescdate,
presc_type,
```

```
item_id as cat_id,
item_name,
med_dosage as quantity,
med_route,
med_form_id,
generic_code,
remarks,
recurrence_daily_id as frequency
from iaim.ip_prescription
order by patient_id;

create temp table base01_ip as
select temp30_5.*, temp100ip.*
from temp100ip
left join temp30_5 on
temp30_5.patient_id = temp100ip.ippatient_id;

/* Services Create Base01_ser */

create temp table base01_ser as
select mr_no,
patient_id,
service_id as cat_id,
presc_date::timestamptz::date as prescdate,
conducted,
conductedby,
conducteddate::timestamptz::date as sercond_date,
prescription_id as consultation_id
from iaim.services_prescribed;

create temp table services as
select service_id as medicine_id, service_name as medicine_name
from iaim.services;
```

```
create temp table med as
select distinct medicine_name, medicine_id
from iaim.medicine_sales_view;

\copy temp30_5 TO 'd:/hospital_data/ProgresSQL/source/pat_diag_vis.csv' CSV HEADER
DELIMITER ',';
\copy base01_ip TO 'd:/hospital_data/ProgresSQL/source/base01_ip.csv' CSV HEADER
DELIMITER ',';
\copy base01_op TO 'd:/hospital_data/ProgresSQL/source/base01_op.csv' CSV HEADER
DELIMITER ',';
\copy base01_ser TO 'd:/hospital_data/ProgresSQL/source/base01_ser.csv' CSV HEADER
DELIMITER ',';

\copy services TO 'd:/hospital_data/ProgresSQL/source/services.csv' CSV HEADER DELIMITER
',';
\copy med TO 'd:/hospital_data/ProgresSQL/source/med.csv' CSV HEADER DELIMITER ',';

/*=========================================*/
/* End of program                          */
/*=========================================*/
```

6.5.2 Primary dataset creation program: 100_adsl.R

```
This R program generates analysis dataset which is used as a primary dataset

#######################################################
# Create calculations using base01_ip and base01_op
#######################################################

#C- Cancelled
#U - Condn. Unnecessary
```

```
#Y -Conducted
#N - Not Conducted
#P - Partially Conducted

library(data.table)
library(dplyr)
library(anytime)

# Get all the data IP, OP and Service

base01_ip <- fread("D:/Hospital_data/ProgresSQL/source/base01_ip.csv")
base01_op <- fread("D:/Hospital_data/ProgresSQL/source/base01_op.csv")
base01_ser <- fread("D:/Hospital_data/ProgresSQL/source/base01_ser.csv")
pat_diag_vis <- fread("D:/Hospital_data/ProgresSQL/source/pat_diag_vis.csv")

# Get the disease category list for MCSD and Metabolic
discat <- data.table( fread ("D:/Hospital_data/ProgresSQL/analysis/discategory.csv") )

# Get the medication and service list
med <- data.table( fread ("D:/Hospital_data/ProgresSQL/source/med.csv") )
ser <- data.table( fread ("D:/Hospital_data/ProgresSQL/source/services.csv") )

medall <- rbind(med, ser, fill = TRUE)
rm(med, ser)

#########################################################
# Work on the services data
# get the date converted to numeric date
# get the minimum and maximum date for each visit
# get the frequency count for each type of service
#########################################################
```

```
base01_ser0 <- base01_ser [,c("mr_no", "patient_id", "prescdate", "sercond_date",
"cat_id", "conducted"), with =FALSE]
base01_ser0 <- base01_ser0 [, `:=` ( newdt = anydate(prescdate),
                                     serdt = anydate(sercond_date) )] [order(mr_no, newdt,
patient_id)]

base01_ser01 <- base01_ser0[, .(serstdt = min(newdt),
                                serendt = max(newdt),
                                freq = .N), by = .(mr_no, patient_id, cat_id, conducted)]

base01_ser01t <- dcast(data = base01_ser01,
                       mr_no + patient_id + cat_id + serstdt + serendt ~ conducted,
                       value.var = c("freq"),
                       fill = "")

base01_ser01t <- merge (x = base01_ser01t,
                        y = medall,
                        by.x = "cat_id",
                        by.y = "medicine_id",
                        all.x = TRUE)

base01_ser01t <- base01_ser01t [order(mr_no, serstdt, patient_id)]
base01_ser01t <- base01_ser01t [, newdt := serstdt]

l = list(IP = base01_ip, OP = base01_op)
base01_all <- rbindlist(l, idcol = "Type", use.names = TRUE, fill = TRUE)

base01_all <- base01_all [, `:=` ( newdt = anydate(prescdate) )] [order(mr_no, newdt,
patient_id)]

################################################
# create visit numbers and total number of visits
# Individual visits: merge the data on base01_all
```

```
# IP visits
# OP visits
# Total number of visits IP + OP
################################################

vis <- unique ( rbind(base01_all [, c("mr_no", "patient_id", "newdt"), with =FALSE],
             base01_ser01t[, c("mr_no", "patient_id", "newdt"), with =FALSE], fill=TRUE
))

vis <- vis [, Type := substr(patient_id, 1, 2)] [order (mr_no, newdt, patient_id)]
vis <- vis [, `:=` (vis =1:.N,
                    all_vis = max( seq_len(.N) ) ), by = .(mr_no)]
vis02 <- vis [, .(vistype =.N), by = .(mr_no, Type, all_vis)]
vis02t <- dcast(data = vis02,
                mr_no +all_vis ~ paste("all_", tolower(Type), sep =""),
                value.var =c("vistype"),
                fill="")
vis03 <- merge (vis [, -c("all_vis")], vis02t, by = "mr_no")

################################################
# Start and end date for each type OP and IP
# Start and end date for overall visit dates
################################################
base01_all01 <- vis[, .(stdt = min(newdt),
                                endt = max(newdt),
                                dur = max(newdt) - min(newdt) + 1), by = .(mr_no, Type)]

base01_all01t <- dcast(data = base01_all01,
                       mr_no ~ Type,
                       value.var = c("stdt", "endt", "dur"),
                       fill = "")

#############################
```

```
# Start for the overall study
############################
base01_all020 <- vis[, .(cstdt = min(newdt),
                         cendt = max(newdt),
                         cdur = max(newdt) - min(newdt) + 1), by = .(mr_no)]

#############################################
# Create one large dataset with all the dates
#############################################
dates_dur <- merge (x = base01_all020,
                    y = base01_all01t,
                    by = c("mr_no"),
                    all.x = TRUE)

vis03dates_dur <- merge (x = dates_dur,
                     y = vis03,
                     by = c("mr_no"),
                     all.x = TRUE)

vis03dates_dur <- vis03dates_dur [, studyday := newdt - cstdt + 1]

################################################
# Merge the Medication information
# Merge the visit information and day calculations
# Merge this information on SERVICEs data as well
################################################

base01_all01 <- merge (x = base01_all,
                       y = medall,
                       by.x = "cat_id",
                       by.y = "medicine_id",
                       all.x = TRUE)
```

```
base01_all011 <- merge (x = base01_all01,
                        y = vis03dates_dur [, -c("Type")],
                        by = c("mr_no", "patient_id", "newdt" ),
                        all.x = TRUE)

################################################
# This should be moved after the VIS calculations
# Add the patient_info
################################################
base01_ser02t <- merge (x = base01_ser01t,
                         y = vis03dates_dur,
                         by = c("mr_no", "patient_id", "newdt" ),
                         all.x = TRUE)

base01_ser02t <- merge (x = base01_ser02t,
                         y = pat_diag_vis,
                         by = c("mr_no", "patient_id"),
                         all.x = TRUE)

all <- rbind(base01_all011, base01_ser02t, fill =TRUE, use.names = TRUE)
all02 <- all [, -c("ippatient_id", "consult_id", "consultation_id" ,"patient_presc_id",
                   "med_form_id", "op_medicine_pres_id", "doctor_id", "diagdate",
                   "prescdate")] [order(mr_no, studyday, patient_id, newdt, vis, cat_id)]

#######################################################
# Calculations for
# Get the disease category list for RMSD and Metabolic
#######################################################
tmpall <- merge (x = discat[, -c("Description"), with =FALSE],
                 y = all02,
                 by.x = "Code",
                 by.y = "icd_code")
```

```
# create a dummy variable
tmpall <- tmpall[ ,val:=1]

subset2 <- tmpall [, c("mr_no", "distype", "val"), with =FALSE]
subset2 <- unique(subset2)

subset3 <- dcast (data = subset2,
                  fill =0,
                  mr_no ~ distype,
                  value.var="val")

# Create an indicator variable to determine
# Both Metabolic and RMSD = 99
# Only Metabolic = 1
# Only RMSD = 2

subset3 <- subset3 [Metabolic == 1 & RMSD == 1, combine := "Metabolic and RMSD"]
subset3 <- subset3 [Metabolic == 1 & RMSD == 0, combine := "Metabolic"]
subset3 <- subset3 [Metabolic == 0 & RMSD == 1, combine := "RMSD"]

all_met_rmsd <- merge (x = subset3,
                       y = all02,
                       by = "mr_no",
                       all.x = TRUE)

all_met_rmsd <- merge (x = discat[, -c("Description" , "date"), with =FALSE],
                   y = all_met_rmsd,
                   all = TRUE,
                   by.x = "Code",
                   by.y = "icd_code")

all_met_rmsd$distype[is.na(all_met_rmsd$distype)] <- "OTHER"
all_met_rmsd <- all_met_rmsd [order(mr_no, studyday, patient_id, newdt, vis, cat_id)]
```

```
# Calculation of first RMSD or Metabolic disease date
minday <- all_met_rmsd[ distype != "OTHER",
                        .(minday = min(studyday)), by =.(mr_no, distype)]
mindayt <- dcast (data = minday,
                  mr_no ~ paste("minday", distype, sep=""),
                  value.var="minday")
all_met_rmsd <- merge (all_met_rmsd, mindayt, by = "mr_no")

# Calculate the age variable for non-missing dates
all_met_rmsd <- all_met_rmsd [, `:=`( age = ifelse ( !is.na( anydate(dateofbirth)) ,
                                                     round( (anydate(newdt) -
anydate(dateofbirth) + 1)/365.25, digits = 0 ), NA),
                                        newdt0 = anydate(newdt)), ]

# Add Indian rutus as new variables
# https://www.drikpanchang.com/seasons/season-tropical-timings.html?geoname-
id=1277333&year=2010

rutus <- fread("D:/Hospital_data/ProgresSQL/analysis/rutus.csv")
rutus <- rutus [, `:=`(startdt = as.POSIXct( startdate, format="%d-%m-%Y"),
                       enddt = as.POSIXct( enddate, format="%d-%m-%Y")) ]

rutus02 <- rutus[ , list(season = season, year = year,
                         newdt0 = anydate( seq(startdt, enddt, by = "day") )), by =
1:nrow(rutus)]

all_met_rmsd <- merge (x = all_met_rmsd,
                       y = rutus02 [, c("newdt0", "year", "season")],
                       by = c("newdt0"),
                       all.x = TRUE)

rm (base01_ip, base01_op, base01_ser, l)
```

```
all_met_rmsd <- all_met_rmsd [, `:=` (baseage = min(age)), by =.(mr_no)]

##############################################
# Update the data by re-coded Medicine names
##############################################
lookup_medicine <- fread("D:/Hospital_data/ProgresSQL/analysis/lookup_medicine.txt",
sep="|")

all_met_rmsd <- merge(x = all_met_rmsd,
                  y = lookup_medicine,
                  all.x = TRUE,
                  by.x = c("medicine_name"),
                  by.y = c("medicine_name")  )

fwrite(all, "D:/Hospital_data/ProgresSQL/analysis/01adsl.csv")
fwrite(all_met_rmsd, "D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.csv")
saveRDS (all_met_rmsd, "D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")

dis_rutu <- all_met_rmsd [Code != "",  .(cnt = uniqueN(mr_no)), by = .(season, Code,
description)] [order(season, -cnt, Code)]
dis_rutu_yr <- all_met_rmsd [Code != "",  .(cnt = uniqueN(mr_no)), by = .(year, season,
Code, description)][order(year, season, -cnt, Code)]
dis_rutu_yr02 <- dcast(dis_rutu_yr,
                       season + Code + description ~ paste("yr", year, sep=""),
                       value.var = c("cnt"),
                       fill=" ")

fwrite(dis_rutu, "D:/Hospital_data/ProgresSQL/analysis/dis_rutu.csv")
fwrite(dis_rutu_yr02, "D:/Hospital_data/ProgresSQL/analysis/dis_rutu_yr.csv")

/*=======================================*/
/* End of program                        */
```

```
/*=======================================*/
```

6.5.3 R and SQL programs for other datasets from SQL database: 02other_data.R

```
This program creates datasets for various CRFs which are not covered in the first
program. This program creates 1 csv file per CRF.  SQL code is added at the top of the
file and then followed by R code

#============================================================
drop table if exists patient_section_details, patient_section_values;
drop table if exists patient_section_details, patient_section_values, base10_other0;

create table patient_section_details as
select mr_no, patient_id, section_id, section_detail_id, section_item_id, item_type
from iaim.patient_section_details
order by mr_no, patient_id;

create table patient_section_values as
select section_detail_id, field_id, option_id, option_remarks
from iaim.patient_section_values
order by section_detail_id;

## Not working
/*
create table base10_other0 as
select patient_section_details.*,
patient_section_values.field_id, patient_section_values.option_id,
patient_section_values.option_remarks
from patient_section_details
full join patient_section_values on
patient_section_details.section_detail_id = patient_section_values.section_detail_id and
patient_section_details.section_item_id = patient_section_values.field_id and
patient_section_details.section_id = patient_section_values.option_id;
```

```
*/

## Working:
create table base10_other11 as
select a.*,
b.field_id, b.option_id, b.option_remarks
from patient_section_details as a, patient_section_values as b where
a.section_detail_id=b.section_detail_id ;

\copy base10_other11 TO 'd:/hospital_data/ProgresSQL/data_chk/base10_other11.csv' CSV
HEADER DELIMITER ',';
\copy iaim.section_master TO 'd:/hospital_data/ProgresSQL/data_chk/section_master.csv'
CSV HEADER DELIMITER ',';
\copy iaim.section_field_options TO
'd:/hospital_data/ProgresSQL/data_chk/section_field_options.csv' CSV HEADER DELIMITER
',';
\copy iaim.section_field_desc TO
'd:/hospital_data/ProgresSQL/data_chk/section_field_desc.csv' CSV HEADER DELIMITER ',';
\copy iaim.patient_consultation_field_values TO
'd:/hospital_data/ProgresSQL/data_chk/patient_consultation_field_values.csv' CSV HEADER
DELIMITER ',';

##############################
Check the _orig dataset

create table patient_section_details_orig as
select mr_no, patient_id, section_id, section_detail_id, section_item_id, item_type
from iaim.patient_section_details
order by mr_no, patient_id;

## Working:
create table base10_other11_orig as
select a.*,
```

```
b.field_id, b.option_id, b.option_remarks
from patient_section_details_orig as a, patient_section_values as b where
a.section_detail_id=b.section_detail_id ;

\copy base10_other11_orig TO
'd:/hospital_data/ProgresSQL/data_chk/base10_other11_orig.csv' CSV HEADER DELIMITER ',';


#============================================================

library(data.table)
library(stringi)
library(stringr)

# Read the data
base01_other <- fread("D:/Hospital_data/ProgresSQL/data_chk/base10_other11.csv")
base01_other02 <- base01_other [nchar(option_remarks)> 0]

# CRF names
section_master <- fread("D:/Hospital_data/ProgresSQL/data_chk/section_master.csv")
section_master <- section_master[, c("section_id", "section_title"), with = FALSE]

base01_other02 <- merge (x = base01_other02,
                         y = section_master,
                         by = "section_id",
                         all.x = TRUE)

# variable names
section_field_options <-
fread("D:/Hospital_data/ProgresSQL/data_chk/section_field_options.csv")

base01_other022 <- merge (x = base01_other02 [ option_id >= 0],
                          y = section_field_options ,
```

```
                        by = c("option_id", "field_id"), #by = c("section_id",
"field_id"),
                        all.x = TRUE)

# Keep Unique records
#base01_other022 <- unique ( base01_other02 [, c("mr_no", "patient_id", "section_id",
"option_id", "field_id", "option_remarks", "section_title", "display_order",
"option_value"), with =FALSE] )
base01_other022 <- unique ( base01_other022 [, c("mr_no", "patient_id", "section_id",
"field_id", "option_remarks", "section_title", "display_order", "option_value"), with
=FALSE] )

# Sort the data by patient and visits
base01_other022 <- base01_other022 [ order(mr_no, patient_id, section_id, field_id,
display_order)]

section_field_desc <-
fread("D:/Hospital_data/ProgresSQL/data_chk/section_field_desc.csv")
section_field_desc <- section_field_desc[, c("section_id", "field_id", "display_order",
"field_name", "no_of_lines"), with = FALSE]

base01_other044 <- merge (x = base01_other02 [ option_id < 0],
                        y = section_field_desc,
                        by = c("section_id", "field_id"),
                        all.x = TRUE)
base01_other044 <- unique ( base01_other044 [, c("mr_no", "patient_id", "section_id",
"option_id", "field_id", "option_remarks", "section_title", "display_order",
"field_name", "no_of_lines"), with =FALSE] )

# Sort the data by patient and visits
base01_other044 <- base01_other044 [ order(mr_no, patient_id, section_id, field_id,
display_order)]
setnames(base01_other044, "field_name", "option_value")
```

```
base01_all <- rbind(base01_other022, base01_other044, fill =TRUE)

#########################################################
# Need to consolidate variable names and combine
# base01_other044
# base01_other022
# Create a counter variable for transposing
#########################################################

# Create a counter variable for transpose
base01_other030 <- base01_all [, subvis := 1:.N, by = .(mr_no, patient_id, section_id,
field_id, option_value)]

#fwrite(base01_other030, "D:/Hospital_data/ProgresSQL/analysis/complete_other_data.csv")


#################################
# Subset for Metabolic RMSD data
#################################
all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")

subpat <- unique(all_met_rmsd [, c("mr_no", "Metabolic", "RMSD", "combine", "all_vis",
                                   "city_name", "state_name", "dateofbirth",
"country_name",
                                   "death_date")])

vispat <- unique(all_met_rmsd [, c("mr_no", "studyday", "patient_id", "newdt", "vis",
"Type", "Code", "distype", "description",
                                   "all_ip", "all_op")])

# Only keep Metabolic and RMSD patients
base01_met_rmsd <- merge (x = base01_other030,
```

```
                            y = subpat [,c("mr_no")],
                            by = c("mr_no"),
                            all.y = TRUE)

sub <- unique( base01_met_rmsd [, c("section_id", "section_title",
                                    "field_id", "display_order", "option_value")] )
[order(section_id, field_id, display_order, option_value)]
sub <- sub [, varnum:=seq_len(.N), by =.(section_id)]
sub <- sub [, trnvar := paste("sec", str_pad(section_id, 3, side = "left", pad = 0),
                              "_var", str_pad(varnum, 3, side = "left", pad = 0),
                              "_", option_value, sep="" )]

base01_met_rmsd <- merge (x = base01_met_rmsd,
                            y = sub,
                            by = c("section_id", "section_title",
                                   "field_id", "display_order", "option_value"),
                            all.x = TRUE)

# Transpose the data as per CRF pages
base01_met_rmsd_trn <- dcast(data = base01_met_rmsd,
                               mr_no + patient_id + subvis ~ trnvar,
                               value.var = c("option_remarks"))

# Add visit information and disease information:
base01_met_rmsd_trn <- merge (x = base01_met_rmsd_trn,
                            y = vispat,
                            by = c("mr_no", "patient_id"),
                            all.x = TRUE)

# Add patient demog + visit + duration information
base01_met_rmsd_trn <- merge (x = base01_met_rmsd_trn,
                                y = subpat,
                                by = c("mr_no"),
```

```
                                  all.x = TRUE)

# Keep variables by section

df = base01_met_rmsd_trn[,(names(base01_met_rmsd_trn) %in%
                             c("mr_no", "patient_id", "Metabolic", "RMSD", "combine",
"subvis",
                               "city_name", "state_name", "dateofbirth", "country_name",
                               "death_date", "Type", "Code", "distype", "description",
                               "studyday", "patient_id", "newdt", "vis", "all_vis",
"all_ip", "all_op")
                           | grepl("^sec004",names(base01_met_rmsd_trn)) ), with =FALSE]

sections <- unique(sub$section_id)
for (ii in sections){

  jj <- str_pad(ii, 3, side = "left", pad = 0)
  kk <- paste0("^sec", jj, sep="")
  print(jj)
  print(kk)

  fwrite(file = paste0("D:/Hospital_data/ProgresSQL/analysis/sec",
                       jj, ".csv"),
         x = base01_met_rmsd_trn [,(names(base01_met_rmsd_trn) %in%
                                      c("mr_no", "patient_id", "Metabolic", "RMSD",
"combine", "subvis","city_name",
                                        "state_name", "dateofbirth", "country_name",
"death_date", "Type", "Code",
                                        "distype", "description", "studyday",
"patient_id", "newdt", "vis", "all_vis",
                                        "all_ip", "all_op")  |
                                      grepl(kk,names(base01_met_rmsd_trn)) ), with
=FALSE]
```

```
  )
}


# dtable <- df[, fwrite(.SD, paste0("./output/"), Name, ".csv"), by = Name]
#############################################################################
# End of program
#############################################################################
```

6.5.4 Analysis program for Figure 3-1

Refer to the 100_adsl.R program

6.5.5 Analysis program for Figure 3-2

R program: 03_typesOfassessment.R

```
library(Hmisc)

library(data.table)

library(stringi)

library(stringr)

library(sqldf)


all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")


#################################

# Subset for Metabolic RMSD data
```

```
##################################
all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")


subpat <- unique(all_met_rmsd [, c("mr_no", "Metabolic", "RMSD", "combine", "all_vis",
"patient_gender", "baseage",
                                   "city_name", "state_name", "dateofbirth",
"country_name",
                                   "death_date")])


vispat <- unique(all_met_rmsd [, c("mr_no", "studyday", "patient_id", "newdt", "vis",
"Type", "Code", "distype", "description",
                                   "all_ip", "all_op")])


################################################
# Get records per visit for Treatment / Procedure
# Med start date, end date non missing and
# name non missing
################################################


med_ip <- unique( na.omit( all_met_rmsd, cols = c("stdt_IP") ))
```

```
med_ip <- unique( med_ip [Type == "IP", c("mr_no", "vis", "studyday", "Metabolic",
"RMSD", "combine", "all_vis", "patient_gender", "baseage"), ] )

med_ip <- med_ip [, cat := "Treatment - IP"]


med_op <- unique( na.omit( all_met_rmsd, cols = c("stdt_OP") ))

med_op <- unique( med_op [Type == "OP", c("mr_no", "vis", "studyday", "Metabolic",
"RMSD", "combine", "all_vis", "patient_gender", "baseage"), ] )

med_op <- med_op [, cat := "Treatment - OP"]


ser <- unique( na.omit( all_met_rmsd, cols = c("serstdt") ))

ser <- unique( ser [, c("mr_no", "vis", "studyday", "Metabolic", "RMSD", "combine",
"all_vis", "patient_gender", "baseage"), ] )

ser <- ser [, cat := "Treatment - Procedure"]


dis <- unique( all_met_rmsd [Code != " " | description != " ", c("mr_no", "vis",
"studyday", "Metabolic", "RMSD", "combine", "all_vis", "patient_gender", "baseage"), ] )

dis <- dis [, cat := "Disease"]


catall <- rbind(med_ip, med_op, ser, dis, fill = TRUE)


# Read the data
```

```
base01_other <- fread("D:/Hospital_data/ProgresSQL/data_chk/base10_other11.csv")
base01_other02 <- base01_other [nchar(option_remarks)> 0]

# CRF names
section_master <- fread("D:/Hospital_data/ProgresSQL/data_chk/section_master.csv")
section_master <- section_master[, c("section_id", "section_title"), with = FALSE]

base01_other02 <- merge (x = base01_other02,
                         y = section_master,
                         by = "section_id",
                         all.x = TRUE)

# variable names
section_field_options <-
fread("D:/Hospital_data/ProgresSQL/data_chk/section_field_options.csv")

base01_other022 <- merge (x = base01_other02 [ option_id >= 0],
                          y = section_field_options ,
                          by = c("option_id", "field_id"), #by = c("section_id",
"field_id"),
                          all.x = TRUE)
```

```
# Keep Unique records

base01_other022 <- unique ( base01_other022 [, c("mr_no", "patient_id", "section_id",
"field_id", "option_remarks", "section_title", "display_order", "option_value"), with
=FALSE] )


# Sort the data by patient and visits

base01_other022 <- base01_other022 [ order(mr_no, patient_id, section_id, field_id,
display_order)]


section_field_desc <-
fread("D:/Hospital_data/ProgresSQL/data_chk/section_field_desc.csv")

section_field_desc <- section_field_desc[, c("section_id", "field_id", "display_order",
"field_name", "no_of_lines"), with = FALSE]


base01_other044 <- merge (x = base01_other02 [ option_id < 0],

                          y = section_field_desc,

                          by = c("section_id", "field_id"),

                          all.x = TRUE)

base01_other044 <- unique ( base01_other044 [, c("mr_no", "patient_id", "section_id",
"option_id", "field_id", "option_remarks", "section_title", "display_order",
"field_name", "no_of_lines"), with =FALSE] )
```

```
# Sort the data by patient and visits
base01_other044 <- base01_other044 [ order(mr_no, patient_id, section_id, field_id,
display_order)]
setnames(base01_other044, "field_name", "option_value")


base01_all <- rbind(base01_other022, base01_other044, fill =TRUE)


#########################################################
# Need to consolidate variable names and combine
# base01_other044
# base01_other022
# Create a counter variable for transposing
#########################################################


# Create a counter variable for transpose
base01_other030 <- base01_all [, subvis := 1:.N, by = .(mr_no, patient_id, section_id,
field_id, option_value)]


# Only keep Metabolic and RMSD patients
base01_met_rmsd <- merge (x = base01_other030,
                          y = subpat [,c("mr_no")],
```

```
                            by = c("mr_no"),
                            all.y = TRUE)

sub <- unique( base01_met_rmsd [, c("section_id", "section_title",
                                    "field_id", "display_order", "option_value")] )
[order(section_id, field_id, display_order, option_value)]
sub <- sub [, varnum:=seq_len(.N), by =.(section_id)]
sub <- sub [, trnvar := paste("sec", str_pad(section_id, 3, side = "left", pad = 0),
                              "_var", str_pad(varnum, 3, side = "left", pad = 0),
                              "_", option_value, sep="" )]

base01_met_rmsd <- merge (x = base01_met_rmsd,
                            y = sub,
                            by = c("section_id", "section_title",
                                   "field_id", "display_order", "option_value"),
                            all.x = TRUE)

base01_met_rmsd02 <- unique( base01_met_rmsd [option_remarks != " ", c("mr_no",
"option_value", "patient_id", "trnvar")] )

# Add visit information and disease information:
```

```
base01_met_rmsd02 <- merge (x = base01_met_rmsd02,
                            y = unique( vispat [, c("mr_no", "patient_id", "vis",
"studyday")] ),
                            by = c("mr_no", "patient_id"),
                            all.x = TRUE,
                           allow.cartesian = TRUE)

# Add patient demog + visit + duration information
base01_met_rmsd02 <- merge (x = base01_met_rmsd02,
                            y = subpat,
                            by = c("mr_no"),
                            all.x = TRUE)

base01_met_rmsd02 <- unique( base01_met_rmsd02 )

# variable names
types <- fread("D:/Hospital_data/ProgresSQL/analysis/lookup_03types.csv")

base01_met_rmsd02 <- merge (x = base01_met_rmsd02,
                           y = types [, c("trnvar", "cat")],
```

```
                            by = c("trnvar"),
                            all.x = TRUE)


base01_met_rmsd03 <- unique( base01_met_rmsd02 [ , -c("option_value", "patient_id",
"trnvar" )])


catall02 <- rbind(catall, base01_met_rmsd03, fill = TRUE)

catall02 <- catall02 [, val :=1]

fwrite(catall02,

       "D:/Hospital_data/ProgresSQL/analysis/03_typesOfassessent.csv")

################################################################################
# End of program
################################################################################
```

6.5.6 Analysis program for Figure 3-3

Refer to the 100_adsl.R program

6.5.7 Analysis program for Figure 3-4

Refer to the 100_adsl.R program

6.5.8 Analysis program for Figure 3-5

Refer to the 100_adsl.R program

6.5.9 Analysis program for Figure 3-6

R program: 04_patients_analysis_tableu_adsl.R

```
library(zoo)
```

```
setwd("C:\\Users\\mahajvi1\\Desktop\\Desktop - copied on 18August2014\\Backup\\Ayur guidelines\\FRLHT\\01 Hospital data 30July2016\\")


# Create Vital sign data from 31st July 2016 version of the data


vitals <- read.csv("Vitals.csv")

vitals <- data.frame ( cbind(vitals, data="vital", type = substr(vitals$Patient.Id, 1, 2) ) )

vitals <- cbind ( data.frame ( subset (vitals, select =c(MR.No., type, data, Age, Gender, City, Country, Blood.Group, First.Visit.Date) )), visdate = vitals$Vital.Date)


Diagnosis <- read.csv("Diagnosis.csv")

Diagnosis <- data.frame ( cbind(Diagnosis,  data="diag", type = substr(Diagnosis$Patient.Id, 1, 2) ) )

Diagnosis <- cbind ( data.frame ( subset (Diagnosis, select =c(MR.No., type, data, Age, Gender, City, Country, Blood.Group, First.Visit.Date, Code) )), visdate = Diagnosis$Admission.Date)


Doctor_consultation <- read.csv("Doctor_consultation.csv")

Doctor_consultation <- data.frame ( cbind(Doctor_consultation, data="Doccon", type = substr(Doctor_consultation$Patient.Id, 1, 2) ) )
```

```
Doctor_consultation <- cbind ( data.frame ( subset (Doctor_consultation, select
=c(MR.No., type, data, Age, Gender, City, Country, Blood.Group, First.Visit.Date) )),
visdate = Doctor_consultation$Cons..Aptmt.Date)


Lab <- read.csv("Lab.csv")

Lab <- data.frame ( cbind(Lab, data="Lab", type = substr(Lab$Patient.Id, 1, 2) ) )

Lab <- cbind ( data.frame ( subset (Lab, select =c(MR.No., type, data, Age, Gender, City,
Country, Blood.Group, First.Visit.Date) )), visdate = Lab$Conducted.Date)


# Combine all the data into 1

all0 <- rbind (vitals, subset(Diagnosis, select =-c(Code)), Doctor_consultation, Lab)

all01 <- data.frame (unique ( subset(all0, select =c(MR.No., type)  ) ))

tmp01op <- data.frame( unique ( subset(all01, select =c(MR.No.), type == "OP")))

tmp01ip <- data.frame( unique ( subset(all01, select =c(MR.No.), type == "IP")))


common <- data.frame ( MR.No.= intersect(tmp01op$MR.No., tmp01ip$MR.No.) )

onlyip <- data.frame ( merge (x = tmp01op, y =tmp01ip, by ="MR.No.", all.y=TRUE) )

onlyop <- data.frame ( merge (x = tmp01ip, y =tmp01op, by ="MR.No.", all.y=TRUE) )


demog02 <- data.frame (unique ( subset (all0, select=-c(data, visdate, First.Visit.Date)
) ) )
```

```
common_demog <- cbind ( data.frame ( merge (x = demog02, y =common, by ="MR.No.",
all.y=TRUE) ), grp="Common")

onlyip_demog <- cbind ( data.frame ( merge (x = demog02, y =onlyip, by ="MR.No.",
all.y=TRUE) ), grp="OnlyIP")

onlyop_demog <- cbind ( data.frame ( merge (x = demog02, y =onlyop, by ="MR.No.",
all.y=TRUE) ), grp="OnlyOP")


all_age02 <- rbind ( common_demog, onlyip_demog, onlyop_demog)

all_age02 <- cbind (all_age02, age2= as.numeric(all_age02$Age) )


rm(list=ls(pattern="vitals"))

#rm(list=ls(pattern="Diagnosis"))

rm(list=ls(pattern="Doctor_"))

rm(list=ls(pattern="Lab"))


tmp02 <- data.frame (cbind (all0,
                            adm = gsub("-", "/", all0$visdate),
                            fdt = gsub("-", "/", all0$First.Visit.Date)
))

tmp02 <- data.frame (cbind (tmp02,
```

```
                            adm2 = as.POSIXct(tmp02$adm, format="%d/%m/%Y"),
                            fdt2 = as.POSIXct(tmp02$fdt, format="%d/%m/%Y")
))

tmp03 <- data.frame (cbind (tmp02,
                            visday = difftime(tmp02$adm2, tmp02$fdt2, units="days") + 1,
                            mondt=  as.yearmon(tmp02$adm2, format="%Y-%m" )
))

tmp04 <- data.frame ( unique ( subset (tmp03, select =c (MR.No., First.Visit.Date,
visdate, adm, fdt, data) )) )

# Count the number of visits per patient
tmp05 <- data.frame (table (tmp04$MR.No.))
tmp05 <- data.frame (cbind (tmp05, MR.No. = tmp05$Var1, Novisits = tmp05$Freq))

# Count the number of diagnosis
tmp06 <- droplevels (unique (data.frame ( subset (Diagnosis, Code !="", select =
c(MR.No., Code ) ) )) )
tmp07 <- subset (data.frame (table (tmp06$MR.No.)), Freq > 0)
tmp07 <- data.frame (cbind (tmp07, MR.No. = tmp07$Var1, Nodiseases = tmp07$Freq))
```

```
# Put all data into 1 with 1 record per patient
final <- data.frame (merge (x= all_age02, y =tmp05, by = "MR.No.", all=TRUE) )
final02 <- data.frame (merge (x= final, y =tmp07, by = "MR.No.", all=TRUE) )
final03 <- data.frame ( subset (final02, select =-c ( type, age2, Freq.x, Freq.y, Var1.x,
Var1.y) ) )
write.csv(final03, file="04_patient_analysis_tableu_adsl.csv")

# Create a table by one record per patient per date
tmp08 <- data.frame (unique ( subset (tmp03, select=c(MR.No., fdt2) ) ) )
tmp09 <- subset (data.frame (table (tmp08$fdt2)), Freq > 0)
tmp09 <- data.frame (cbind (tmp09, Newpatients = tmp09$Freq))
tmp09 <- data.frame ( subset (tmp09, select =-c (Freq) ) )

# Create 1 record per date for number of patients on a day
tmp10 <- data.frame (unique ( subset (tmp03, select=c(MR.No., adm2) ) ) )
tmp11 <- subset (data.frame (table (tmp10$adm2)), Freq > 0)
tmp11 <- data.frame (cbind (tmp11, Visitpatients = tmp11$Freq))
tmp11 <- data.frame ( subset (tmp11, select =-c (Freq) ) )
```

```
# Create 1 record per date per patient type for first visit date
tmp12 <- data.frame (unique ( subset (tmp03, select=c(MR.No., fdt2, type) ) ) )
tmp13 <- subset (data.frame (table (tmp12$fdt2, tmp12$type)), Freq > 0)
tmp13_tran <- reshape (tmp13,
                       direction="wide",
                       idvar= c("Var1"),
                       timevar="Var2" )
tmp13_tran <- data.frame (cbind (tmp13_tran, newIP = tmp13_tran$Freq.IP, newOP =
tmp13_tran$Freq.OP))
tmp13_tran <- data.frame ( subset (tmp13_tran, select =-c (Freq.IP, Freq.OP) ) )



# Create 1 record per date per patient type for other visit dates
tmp14 <- data.frame (unique ( subset (tmp03, select=c(MR.No., adm2, type) ) ) )
tmp15 <- subset (data.frame (table (tmp14$adm2, tmp14$type)), Freq > 0)
tmp15_tran <- reshape (tmp15,
                      direction="wide",
                      idvar= c("Var1"),
                      timevar="Var2" )
tmp15_tran <- data.frame (cbind (tmp15_tran, visitsIP = tmp15_tran$Freq.IP, visitsOP =
tmp15_tran$Freq.OP))
```

```
tmp15_tran <- data.frame ( subset (tmp15_tran, select =-c (Freq.IP, Freq.OP) ) )

# Create a table by one record per patient per date for determining what measurements
# are done on which dates
tmp16 <- data.frame (unique ( subset (tmp03, select=c(MR.No., adm2, data) ) ) )
tmp17 <- subset (data.frame (table (tmp16$adm2, tmp16$data)), Freq > 0)
tmp17_tran <- reshape (tmp17,
                       direction="wide",
                       idvar= c("Var1"),
                       timevar="Var2" )

finalcal <- data.frame (merge (x= tmp09, y =tmp11, by = "Var1", all=TRUE) )
finalcal02 <- data.frame (merge (x= finalcal, y =tmp13_tran, by = "Var1", all=TRUE) )
finalcal03 <- data.frame (merge (x= finalcal02, y =tmp15_tran, by = "Var1", all=TRUE) )
finalcal04 <- data.frame (merge (x= finalcal03, y =tmp17_tran, by = "Var1", all=TRUE) )
write.csv(finalcal04, file="04_patient_analysis_tableu_calendar.csv")

# Create 1 record per patient per disease with all other variables
# This will help us understand age group as well as other demog factors
# for different diseases
```

```
dis <- droplevels (unique (data.frame ( subset (Diagnosis, Code !="", select = c(MR.No., Code ) ) )) )

dis02 <- data.frame (merge (x= dis, y =all_age02, by = "MR.No.", all.x=TRUE) )

write.csv(dis02, file="04_patient_analysis_disease.csv")


# Create 1 record per patient per date per diagnosis

# Print this using the calendar display

# Display this with patient ID as a filter

dis03 <- droplevels (unique (data.frame ( subset (Diagnosis, Code !="", select = c(MR.No., Code, visdate ) ) )) )

dis04 <- data.frame (merge (x= dis03, y =all_age02, by = "MR.No.", all.x=TRUE) )

write.csv(dis04, file="04_patient_analysis_disease_cal.csv")

################################################################################
# End of program
################################################################################
```

6.5.10 Analysis program for Figure 3-7

Refer to program: 04_patients_analysis_tableu_adsl.R

6.5.11 Analysis program for Figure 3-8

Refer to program: 04_patients_analysis_tableu_adsl.R

6.5.12 Analysis program for Figure 3-9

Refer to program: 04_patients_analysis_tableu_adsl.R

6.5.13 Analysis program for Figure 3-10

Refer to program: 04_patients_analysis_tableu_adsl.R

6.5.14 Analysis program for Figure 3-11

Refer to program: 04_patients_analysis_tableu_adsl.R

6.5.15 Analysis program for Figure 3-12

R program: rmsd_metabolic_all.R

```
######################

## Call all programs

######################


source("C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\prgm\\rmsd_metabolic_subset.R")

source("C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\prgm\\diagnosis_primary.R")

source("C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\prgm\\diagnosis_primary_month.R")

source("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\prgm\\diagnosis.R")

source("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\prgm\\vital_sign.R")

source("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\prgm\\lab.R")
```

```
## Calculate all the common variables from all datasets
## merge them back onto the individual datasets

keep <- c("MRNo", "Age", "AgeIn", "Gender", "City", "Country",
          "Bloodgrp", "data", "type", "visday")

# Combine all datasets
nvisall <- data.table ( rbind (diag2 [, keep, with = FALSE],
                               vitals2[, keep, with = FALSE],
                               lab2   [, keep, with = FALSE][MRNo !=""] ) )
# Add a dummry variable
nvisall <- nvisall[, cal :=1]

# Unique patient values
n1unq <- unique ( nvisall [, c("MRNo", "Age", "AgeIn", "Gender", "City","Country"), with
= FALSE] )

# Count distinct visits for each domain based on diag2, vitals2 and lab2:
n1vis <- nvisall[ , .(novis = uniqueN(visday) ), by = .(MRNo, data) ]
n2vis <- dcast(n1vis, MRNo ~ data, value.var = "novis")
```

```
# No of diseases
n1dis <- unique ( diag2 [, c("MRNo", "noofdis"), with = FALSE] )

# Blood group creation
n1blood <- unique( nvisall [ Bloodgrp != "", c("MRNo", "Bloodgrp"), with =FALSE])

# OP, IP creation
n1type <- unique (nvisall [ type != "", c("MRNo", "type", "cal"), with =FALSE])
n2type <- dcast(n1type, MRNo ~ type, value.var = "cal")

# Combine the diag2 dataset and discat data and
# determine diseases which are in RMSD and Metabolic categories vs. OTHER categories
subset5 <- merge (x = discat[, -c("Description"), with =FALSE],
                  y = diag2[, -c("Code"), with =FALSE],
                  all = TRUE,
                  by.x = "Code",
                  by.y = "Code2")

subset5 <- subset5 [,-c("Age", "AgeIn", "Gender", "City", "Country",
```

```
                        "Bloodgrp", "data", "type", "noofdis"), with =FALSE]

# Code non metabolic and non RMSD diseases as OTHER
subset5$distype[is.na(subset5$distype)] <- "OTHER"

# Count number of diseases and freq count for each patient
# ??????????????????????????????????????????????????
# ???? Understand why Code2 is not coming out correctly
# ??????????????????????????????????????????????????
subset6 <- subset5[, .(cnt = uniqueN(Code),
                       frq = .N), by =.(MRNo, distype)]

# Transpose the data and get in 1 row per patient
subset7 <- dcast(subset6,
                 MRNo ~ distype,
                 value.var = c("cnt", "frq"),
                 fill = 0)

# Combine all variables into 1 dataset ADSL
adsl <- Reduce(function(...) merge(..., all = TRUE, by = "MRNo"),
```

```
                list(n1unq, n1dis, n1blood, n2type, n2vis, subset7))

setnames(adsl, "vital", "nvis_vital")
setnames(adsl, "diag", "nvis_diag")
setnames(adsl, "lab", "nvis_lab")

#########################################################
# Create a subsetted data for RMSD and Metabolic diseases
# For analysis create a few additional variables
#########################################################

adsl_sub <- Reduce(function(...) merge(..., all.x = TRUE, by = "MRNo"),
                   list(subset3, adsl))

# Merge this information onto diag2 dataset
diag8 <- Reduce(function(...) merge(..., all.x = TRUE, by = "MRNo"),
                   list(adsl_sub, subset5 ))
write.csv(diag8, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_diag.csv", na=" ")
```

```
# Merge this information onto vitals5 dataset
vitals8 <- Reduce(function(...) merge(..., all.x = TRUE, by = "MRNo"),
                  list(adsl_sub, vitals5 ))
write.csv(vitals8, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_vital.csv", na=" ")

# http://stackoverflow.com/questions/21560500/data-table-merge-based-on-date-ranges
# Version 2 of the code

#########################################################
# Create a vertical vesion of diag_vital
#########################################################
vitals5[, tempday := visday]  ## Add a redundant day column to use as the end range
setkey(diag2, MRNo, min, max) ## Set the key for patient IDs ("y" table)

## Find the overlaps, remove the redundant lossDate2 column, and add the inPolicy column:
ans2 <- foverlaps(vitals5,
                  diag2,
                  by.x=c("MRNo", "visday", "tempday"))[, `:=`(inPolicy=T, tempday=NULL)]
```

```
## Update rows where the claim was out of policy:
ans2[is.na(min), inPolicy:=F]

## Remove duplicates (such as policyNumber==123 & claimNumber==3),
##  and add policies with no claims (policyNumber==125):
setkey(ans2, MRNo,Code2,visday, min) ## order the results
setkey(ans2, MRNo, Code2) ## set the key to identify unique values
ans2 <- rbindlist(list(
  ans2, ## select only the unique values
  diag2[!.(ans2[, unique(MRNo)])] ## policies with no claims
), fill=T)

ans20 <- ans2 [, -c("Age", "AgeIn", "City", "Country", "Bloodgrp",
                    "Gender","noofdis"), with = FALSE]

ans3 <- Reduce(function(...) merge(..., all.y = TRUE, by = "MRNo"),
               list(ans20, adsl_sub))
```

```
write.csv(ans3, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_vital_vertical.csv", na=" ")


#########################################################
# Create primary disease data, combination of 1 disease
# considered as primary disease and display all other
# diseases
#########################################################

prim_diag2 <- Reduce(function(...) merge(..., all.y = TRUE, by = "MRNo"),
                     list(prim_diag, adsl_sub))


write.csv(prim_diag2, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_primary_diag.csv", na=" ")


#########################################################
# Create primary disease data, combination of 1 disease
# considered as primary disease and display all other
# diseases
#########################################################
```

```
prim_diag_mon2 <- Reduce(function(...) merge(..., all.y = TRUE, by = "MRNo"),
                         list(prim_diag_mon, adsl_sub))

write.csv(prim_diag_mon2, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_primary_diag_mon.csv", na=" ")

# Create a lookup table to get the cumulative view of the patients
# Merge this onto individual datasets to create multiple records for patients
# Use vismon variable and create many to many join

dur    <- c(">=1 day", ">=1 month", ">=2 months", ">=3 months", ">=6 months",
            ">=1 year", ">=2 years", ">=3 years", ">=4 years", ">=5 years")

durlwr <- c(0,           1,           2,           3,            6,
            12,          24,          36,           48,           60)

durupr <- c(999,         999,         999,         999,          999,
            999,         999,          999,         999,          999)

ref <- data.table ( cbind.data.frame (durlwr, durupr, dur ) )
```

```
# http://stackoverflow.com/questions/21560500/data-table-merge-based-on-date-ranges
# Version 2 of the code

## The foverlaps function requires both tables to have a start and end range,
# and the "y" table to be keyed
diag8[, tempmon := vismon]  ## Add a redundant day column to use as the end range
setkey(ref, durlwr, durupr) ## Set the key for patient IDs ("y" table)

## Find the overlaps, remove the redundant lossDate2 column, and add the inPolicy column:
diag8rpt <- foverlaps(diag8,
                      ref,
                      by.x=c("vismon", "tempmon"))[, `:=`(inPolicy=T, tempmon=NULL)]

## Update rows where the claim was out of policy:
diag8rpt[is.na(durlwr), inPolicy:=F]

## Remove duplicates (such as policyNumber==123 & claimNumber==3),
##  and add policies with no claims (policyNumber==125):
setkey(diag8rpt, MRNo, Code, vismon, durlwr) ## order the results
```

```
setkey(diag8rpt, MRNo, Code, dur) ## set the key to identify unique values
diag8rpt <- rbindlist(list(
  diag8rpt, ## select only the unique values
  diag8[!.(diag8rpt[, unique(MRNo)])] ## policies with no claims
), fill=T)

# Count number of unique patients, with only 1 visit, 2 visits,3 visits, etc.
diag9rpt <- diag8rpt [, unqvisit := uniqueN(dur), by = .(MRNo)] [order(MRNo, durlwr)]
diag10rpt <- diag9rpt [, .(nopat = uniqueN(MRNo)), by = .(unqvisit)] [order(unqvisit)]

# Create a file for
write.csv(diag9rpt, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_diag_repeat.csv", na=" ")

# Cumulative view on the vital signs data

vitals8[, tempmon := vismon]  ## Add a redundant day column to use as the end range
setkey(ref, durlwr, durupr) ## Set the key for patient IDs ("y" table)

## Find the overlaps, remove the redundant lossDate2 column, and add the inPolicy column:
```

```
vitals8rpt <- foverlaps(vitals8 [vismon > 0],
                        ref,
                        by.x=c("vismon", "tempmon"))[, `:=`(inPolicy=T, tempmon=NULL)]

## Update rows where the claim was out of policy:
vitals8rpt[is.na(durlwr), inPolicy:=F]

## Remove duplicates (such as policyNumber==123 & claimNumber==3),
##  and add policies with no claims (policyNumber==125):
setkey(vitals8rpt, MRNo, vitalparam, vismon, durlwr) ## order the results
setkey(vitals8rpt, MRNo, vitalparam, dur) ## set the key to identify unique values
vitals8rpt <- rbindlist(list(
  vitals8rpt, ## select only the unique values
  vitals8[!.(vitals8rpt[, unique(MRNo)])] ## policies with no claims
), fill=T)

# Create a file for vital signs observations repeated for
# cumulative time point
write.csv(vitals8rpt, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_vital_vertical_repeat.csv", na=" ")
```

```
# Create primary diagnosis and all other diagnosis view by year
# This gives a view about patient having many diseases simultaneously
# along with other diseases

setkeyv (diag8rpt, c("MRNo", "Code", "Description", "dur", "durlwr"))
diag30 <- unique(diag8rpt)

diag40 <- diag30[, `:=`(primarycode = Code,
                        primarydesc = Description,
                        primarydur  = dur,
                        primarydurlwr = durlwr),]

diag50 <- diag40[, c("MRNo", "Age", "AgeIn", "Gender", "City","Country", "Bloodgrp",
                     "Code","distype", "Description", "dur", "durlwr", "durupr",
                     "Metabolic", "RMSD", "combine"), with =FALSE]

diag60 <- diag40[, c("MRNo", "primarycode", "primarydesc",
                     "primarydur", "primarydurlwr"), with =FALSE]
```

```
# set the ON clause as keys of the tables:
setkey(diag50,MRNo, dur)
setkey(diag60,MRNo, primarydur)

# perform the join
prim_diag_dur_repeat <- data.table( merge(diag50,diag60,
                                          all=TRUE,
                                          allow.cartesian = TRUE) )

# Clean up some space
rm (list = ls( pattern = "lab*") )
rm (list = ls( pattern = "n1*") )

# Create a file for vital signs observations repeated for
# cumulative time point
write.csv(prim_diag_dur, file ="C:\\Users\\Lucky\\Documents\\Hospital
data\\01_31JUL2016\\analysis\\rmsd_met_primary_diag_dur_repeat.csv", na=" ")

# Find difference between 2 consecutive visits
# This may give us some idea about the data
```

```
diff <- diag8 [, c("MRNo", "visday"), with =FALSE]

setkey(diag8, MRNo, visday)

diff <- unique(diff) [order (MRNo, visday)]

diff <- diff[,diff:=c(NA,diff(visday)),by=MRNo]

summary(diff$diff)

################################################################################
# End of program
################################################################################
```

6.5.16 Analysis program for Figure 3-13

Refer to program: rmsd_metabolic_all.R

6.5.17 Analysis program for Figure 3-14

Refer to program: rmsd_metabolic_all.R

6.5.18 Analysis program for Figure 3-15

Refer to program: rmsd_metabolic_all.R

6.5.19 Analysis program for Figure 3-16

R program: 085_dis_1st_time_refCal_NodesEdges.R

```
library(data.table)

library(stringi)

library(stringr)

library(sqldf)
```

```
all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")

unqdis <- unique( all_met_rmsd [, c("mr_no", "studyday", "Code", "description"),])
unqdis <- unqdis [!Code %in% c("" , " ")]
unqdis <- unqdis [, mindisday := min(studyday), by = .(mr_no, Code, description)]
unqdis <- unique( unqdis [, c("mr_no", "mindisday", "Code", "description"),])

setnames(unqdis, "Code", "refcode")
setnames(unqdis, "description", "refdesc")

##################################################
# Create this data with min refday for each disease
##################################################

all_met_rmsd02 <- merge(x = all_met_rmsd,
                        y = unqdis,
                        by = c("mr_no"),
                        allow.cartesian = TRUE)
```

```
all_met_rmsd02 <- all_met_rmsd02 [, c("mr_no", "Code", "description", "combine", "RMSD",
"Metabolic",
                                    "newdt0", "Type_med", "Coded_med", "studyday",
"mindisday",
                                    "refcode", "refdesc", "patient_gender", "age",
"baseage",
                                    "distype", "cdur"), ]

#############################################################
# Calculate reference day for each disease as before and after
# studyday and mindisday
#############################################################
all_met_rmsd02 <- all_met_rmsd02 [, refday := ifelse(studyday >= mindisday,
                                                     studyday - mindisday + 1,
                                                     studyday - mindisday),]

all_met_rmsd02 <- all_met_rmsd02[, refmnyr := ifelse(refday >= 1,
                                                     as.numeric( ceiling (refday /
30.4375) ),
                                                     as.numeric( floor (refday / 30.4375)
) ), ]
```

```
period01 <- fread("D:/Hospital_data/ProgresSQL/analysis/lookup_1st_nodesedges.csv")
period02 <- period01[ , list(period = period, periodn = periodn,
                          refmnyr = seq(as.numeric(start), as.numeric(end)) ), by =
1:nrow(period01)]

all_met_rmsd02 <- merge (x = all_met_rmsd02,
                       y = period02 [, c("refmnyr", "period", "periodn")],
                       by = c("refmnyr"),
                       all.x = TRUE)

###################################
# Post process to get day 1 as Day 1
###################################
all_met_rmsd02 <-  all_met_rmsd02[, period := ifelse(refday == 1, 1, period), ]
all_met_rmsd02 <-  all_met_rmsd02[, periodn := ifelse(refday == 1, "Day 1", periodn), ]

saveRDS (all_met_rmsd02, "D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")

disease <- unique(all_met_rmsd02 [, -c("Type_med", "Coded_med"),])
disease <- disease [, cat :="Disease"]
```

```
meds <- unique(all_met_rmsd02 [, -c("Code", "description"),])
meds <- meds [, cat :="Medicine"]
meds <- meds [, Coded_med := paste(Type_med, Coded_med, sep=":"),]

setnames(meds, "Type_med", "Code")
setnames(meds, "Coded_med", "description")

all <- rbind(disease, meds)

bfraftr <- all [, .(min = min(refday), max = max(refday)), by = .(mr_no, refcode,
refdesc, Code, description)]

bfraftr <- sqldf("select *,
                  case
                   When min <= 0 and max <= 0 then 'Reported only before'
                   When min >= 1 and max >= 1 then 'Reported on or after'
                   When min <= 0 and max >= 1 then 'Reported before and after'
                  end as classification
                  from bfraftr")
```

```
all02 <- merge (x = all,
                y = bfraftr ,
                by = c("mr_no", "refcode", "refdesc", "Code", "description"),
                all.x = TRUE)
all02 <- all02 [, -c("min", "max"),]
fwrite(all02,
       "D:/Hospital_data/ProgresSQL/analysis/085_dis_1st_time_refCal_NodesEdges.csv")
saveRDS (all02,
"D:/Hospital_data/ProgresSQL/analysis/085_dis_1st_time_refCal_NodesEdges.rds")

##############################################
# Save disease and medicine version of the data
##############################################
all_met_rmsd02 <- merge (x = all_met_rmsd02,
                         y = bfraftr ,
                         by = c("mr_no", "refcode", "refdesc", "Code", "description"),
                         all.x = TRUE)
all_met_rmsd02 <- all_met_rmsd02 [, -c("min", "max"),]
saveRDS (all_met_rmsd02, "D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")
```

```
################################################################################
# End of program
################################################################################
```

6.5.20 Analysis program for Figure 3-17

R program: 060_allopathic_diag.R

```
library(dplyr)

library(data.table)

library(fuzzyjoin)

library(stringr)

library(stringi)

library(stringdist)

library(quanteda)

library(tm)

library(tidyr)

library(sqldf)


#####################################################################

# This section creates the allopathic diagnosis as per ICD 10 dictionary

#####################################################################

all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")
```

```
all_met_rmsd <- all_met_rmsd [, `:=` (baseage = min(age)), by =.(mr_no)]
all_met_rmsd <- all_met_rmsd [, `:=` (vismon = round( cdur/30.4375, digits = 0))]

# Baseline age
age01 <- unique( all_met_rmsd [, c("mr_no", "baseage", "patient_gender", "cdur")] )

lookup_allopathic_diag <-
fread("D:/Hospital_data/ProgresSQL/analysis/lookup_allopathic_diag.txt", sep="|")

chkpat <- function (dname, var, dataout = "unqsec") {
  sec <- readRDS( paste("D:/Hospital_data/ProgresSQL/analysis/", noquote(dname) , ".rds",
sep="") )


  sec011 <- sec [, `:=` (orig = get(var),
                         all_diag = toupper( get( var))), ]

  ##########################################################
  # Replace multiple diseases in 1 row to multiple rows,
  # Seperate_rows allows: keeping all other rows as is
```

```
  ##########################################################
  sec011_1 <- separate_rows(sec011, all_diag, sep =",|\r\n|\\?|\\bAND\\b|;" )
  #sec011_1 <- sec011_1 [, all_diag := trimws(all_diag), ]
  #sec011_1 <- sec011_1 [, dname := paste(var, sep = ""), ]

  sec011_1 [, all_diag := trimws(all_diag)]
  sec011_1 [, dname := paste(var, sep = "")]

  ##########################################################################
  # Create unique row per disease, this will be matched against ICD 10
  ##########################################################################
  #unqsec <- unique( sec011_1 [, c("all_diag", "dname"), ] )
  tmp <- sec011_1

  assign(dataout, tmp, envir=.GlobalEnv)
}

chkpat(dname = "sec011", var= "sec011_var001_Allopathic Diagnosis", dataout = "dsec011")
chkpat(dname = "sec082", var= "sec082_var001_Allopathic Diagnosis", dataout = "dsec082")
chkpat(dname = "sec122", var= "sec122_var001_Allopathic Diagnosis", dataout = "dsec122")
```

```
chkpat(dname = "sec123", var= "sec123_var001_Allopathic Diagnosis", dataout = "dsec123")

dislist <- lapply(ls(pattern="dsec*"), get)
dis_all <- data.table( rbindlist (dislist))

dis_all02 <- merge(x = dis_all,
                   y = lookup_allopathic_diag,
                   all.x = TRUE,
                   allow.cartesian=TRUE,
                   by = c("all_diag", "dname")  )

##########################
# Subset for coded records
##########################
dis_pat <- dis_all02 [ nchar(code01) > 0 ]
dis_pat <- dis_pat[ code01 != c("** Can not be coded")]

dis_pat02 <- merge (x = dis_pat,
                    y = age01,
                    by = c("mr_no"))
```

```
unqpat <- dis_pat02 [, .(npat = uniqueN (mr_no)), by = .(combine)]
unqpat_gender <- dis_pat02 [, .(npat = uniqueN (mr_no)), by = .(combine, patient_gender)]



chk01 <- unique( dis_pat02 [, c("mr_no", "code01", "text01", "baseage",
                                "patient_gender", "combine", "Metabolic", "RMSD", "cdur",
"all_vis")])

chk01 <- chk01 [, high := substr(code01, 1, 3)]

#################
# ICD dictionary
#################
icd10 <- fread("D:/Hospital_data/ProgresSQL/analysis/icd10cm_order_2018.csv", header=
FALSE)
cats <- data.table( expand.grid( cat1 = LETTERS,
                                 cat2 = seq (0, 99) ) )
cats <- cats [, high := paste( cat1, str_pad(cat2, 2, side = "left", pad = 0), sep=""), ]

cats <- sqldf("select *,
```

```
            case
            When cat1 == 'A' OR cat1 == 'B' then 'Certain infectious and parasitic
diseases'
            When (cat1 == 'C' OR (cat1 == 'D' AND cat2 < 50)) then 'Neoplasms'
            When (cat1 == 'D' AND cat2 >=50) then 'Diseases of the blood and blood-
forming organs and certain disorders involving the immune mechanism'
            When (cat1 == 'E') then 'Endocrine, nutritional and metabolic diseases'
            When (cat1 == 'F') then 'Mental and behavioural disorders'
            When (cat1 == 'G') then 'Diseases of the nervous system'
            When (cat1 == 'H' and cat2 <= 59) then 'Diseases of the eye and adnexa'
            When (cat1 == 'H' and cat2 > 59) then 'Diseases of the ear and mastoid
process'
            When (cat1 == 'I') then 'Diseases of the circulatory system'
            When (cat1 == 'J') then 'Diseases of the respiratory system'
            When (cat1 == 'K') then 'Diseases of the digestive system'
            When (cat1 == 'L') then 'Diseases of the skin and subcutaneous tissue'
            When (cat1 == 'M') then 'Diseases of the musculoskeletal system and
connective tissue'
            When (cat1 == 'N') then 'Diseases of the genitourinary system'
            When (cat1 == 'O') then 'Pregnancy, childbirth and the puerperium'
            When (cat1 == 'P') then 'Certain conditions originating in the perinatal
period'
```

```
              When (cat1 == 'Q') then 'Congenital malformations, deformations and
chromosomal abnormalities'

              When (cat1 == 'R') then 'Symptoms, signs and abnormal clinical and
laboratory findings, not elsewhere classified'

              When (cat1 == 'S' OR cat1 == 'T') then 'Injury, poisoning and certain other
consequences of external causes'

              When (cat1 == 'V' OR cat1 == 'W' OR cat1 == 'X' OR cat1 == 'Y') then
'Injury, poisoning and certain other consequences of external causes'

              When (cat1 == 'Z') then 'Factors influencing health status and contact with
health services'

              When (cat1 == 'U') then 'Codes for special purposes'

              end as icd

              from cats")


##############################################

# Get the 2nd level terms from ICD dictionary

##############################################

icd_sub <- icd10 [ V2 %in% unique (chk01$high)]


#############################################################

# Merge the high level terms and 2nd level terms with the data

#############################################################
```

```
chk02 <- merge (x = chk01,
                y = cats,
                all.x = TRUE,
                by = c("high"))


chk02 <- merge (x = chk02,
                y = icd_sub [, c("V2", "V5")],
                all.x = TRUE,
                by.x = c("high"),
                by.y = c("V2"))


fwrite(chk02,
       "D:/Hospital_data/ProgresSQL/analysis/060_allopathic_diag.csv")
################################################################################
# End of program
################################################################################
```

6.5.21 Analysis program for Figure 3-18

Refer to R program: 060_allopathic_diag.R

6.5.22 Analysis program for Figure 3-19

Refer to R program: 060_allopathic_diag.R

6.5.23 Analysis program for Figure 3-20

Refer to R program: 060_allopathic_diag.R

6.5.24 Analysis program for Figure 3-21

Refer to R program

6.5.25 Analysis program for Figure 3-22

R program: diagnosis_primary.R

```
################################

# Code to generate DIAGNOSIS data

################################

library(data.table)

setwd ("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\source")


diag <- fread("Diagnosis.csv", check.names = FALSE)

diag2 <- diag[, c(1, 8, 9), with=FALSE ]

setnames(diag2, "MR No.", "MRNo")


# Code ** NOT YET CODED for the missing code values

# Sort the data by patient and day

diag2 <- diag2[, Code2 := ifelse(Code ==="", "ZZZ999", Code), ] [order(MRNo, Code2)]

setkeyv (diag2, c("MRNo", "Code2", "Description"))

diag3 <- unique(diag2)
```

```
diag4 <- diag3[, `:=`(primarycode = Code2,
                      primarydesc = Description),]


diag3 <- diag3[, c(1, 3, 4), with =FALSE]
diag4 <- diag4[, c(1, 5, 6), with =FALSE]


# set the ON clause as keys of the tables:
setkey(diag3,MRNo)
setkey(diag4,MRNo)


# perform the join
prim_diag <- merge(diag3,diag4, all=TRUE, allow.cartesian = TRUE)
################################################################################
# End of program
################################################################################
```

6.5.26 Analysis program for Figure 3-23

Refer to R program: diagnosis_primary.R

6.5.27 Analysis program for Figure 3-24

Refer to R program: diagnosis_primary.R

6.5.28 Analysis program for Figure 3-25

R program: diagnosis_primary_mon.R

```
################################
# Code to generate DIAGNOSIS data
################################

library(data.table)
setwd ("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\source")

diag <- fread("Diagnosis.csv", check.names = FALSE)
diag2 <- diag[, c(1, 8, 9, 10), with=FALSE ]
setnames(diag2, "MR No.", "MRNo")
setnames(diag2, "Admission Date", "visdate")

# Code ** NOT YET CODED for the missing code values
# Sort the data by patient and day
# Create date variables and find the difference
diag2 <- diag2[, visdate := as.POSIXct( gsub("-", "/", visdate), format="%d/%m/%Y") ]
diag2 <- diag2[, Code2 := ifelse(Code =="", "ZZZ999", Code), ] [order(MRNo, Code2)]
```

```
# Only extract month part
diag2 <- diag2[, month := format(as.Date(visdate), "%m"), ]

setkeyv (diag2, c("MRNo", "Code2", "Description", "month"))
diag3 <- unique(diag2)

diag4 <- diag3[, `:=`(primarycode = Code2,
                      primarydesc = Description,
                      primarymon  = month),]

diag5 <- diag4[, c(1, 3, 5, 6), with =FALSE]
diag6 <- diag4[, c(1, 7, 8, 9), with =FALSE]

# set the ON clause as keys of the tables:
setkey(diag5,MRNo, month)
setkey(diag6,MRNo, primarymon)

# perform the join
prim_diag_mon <- merge(diag5,diag6, all=TRUE, allow.cartesian = TRUE)
##############################################################################
```

```
# End of program
#########################################################################################
```

6.5.29 Analysis program for Figure 3-26

```
R program:  085_dis_counts_bruce_java.R


#####################################################################
# This is used for 085_dis_count_edges_3rd_byPeriod Tableau display
#####################################################################

library(data.table)
library(stringi)
library(stringr)
library(sqldf)
library(tidyr)
library(rjson)
library(jsonlite)
library(dplyr)

#####################################################################
# These 2 are created using 085_dis_1st_time_refCal_NodeEdges.R program
```

```
#####################################################################

all_met_rmsd02 <- readRDS ("D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")
all_met_rmsd02 <- all_met_rmsd02 [, Coded_med := paste(Type_med, Coded_med, sep=":"),]

all_met_rmsd02 <- all_met_rmsd02 [, Coded_med := str_replace_all(Coded_med, "\"", ""),]

chk01 <- all_met_rmsd02 [, .(cnt = uniqueN(mr_no)),
                         by = .(refcode, refdesc, Code, description, Type_med, Coded_med
)]
chk01 <- chk01[Code != "" & Coded_med != ""]
chk01 <- chk01 [, `:=` (Code02 = paste(Code, ":", description, "->", Coded_med, sep =""),
                        name = paste(refcode, refdesc, sep =","),
                        key = paste(Code, description, sep=",") ), ]

med <- unique( chk01 [Coded_med != c("", " "), c("Coded_med"), ])
setnames(med, "Coded_med", "name")
dis <- unique( chk01 [name != c("", " "), c("name"), ])
meddis <- rbind(med, dis)
meddis <- meddis [, nrow := .I,]
```

```
#####################################################################
# Create a version of data as follows:
# Fixed nodes as relation between
# (1) Period + Reference disease <--> other diseases
# (2) other diseases <--> Medicine
# Use the other diseases section for creating the moving nodes
#####################################################################
part01 <- chk01 [, c("name", "key", "Code02", "cnt", "refdesc", "refcode"), ]

part02 <- chk01 [, c("Coded_med", "key", "Code02", "cnt", "refdesc", "refcode"), ]
setnames(part02, "Coded_med", "name")

part03 <- rbind (part01, part02)
part03 <- merge(part03,
                meddis,
                by = c("name"),
                all = TRUE)

part03 <- part03 [, num := .N, by =.(refdesc, key)]
```

```
part03 <- part03 [, maxnum := max(num), by =.(refdesc)]
part03 <- part03 [, pernum := (num / maxnum) * 100,]

#####################################################################
# Create the Json file
#
#[{
#  "name": "addons",
#  "count": 1,
#  "key": "addons",
#  "pages": [{
#    "name": "A year in apps script and my bucket list",
#    "key": "44784597234089306 41",
#    "title": "A year in apps script and my bucket list",
#    "url": "http://excelramblings.blogspot.com/2015/01/a-year-in-apps-script-and-my-
bucket-list.html"
# }]
# }
#
# "name" : use key
```

```
# "count" : use num
# "key" : use key
# "pages" :
#   "names" : use name
#   "key" : use nrow
#   "title" : use name
#   "url" : Code02
#######################################################################

part04 <- part03 [, frstprt := paste('{"name" :"', key, '", "count" :', pernum, ', "key"
:"', key, '",', sep ="" ), ]
part04 <- part04 [, scndprt := paste('{"name" :"', name, '", "key" :', nrow, ', "title"
:"', name, '", "url" :"', Code02, '"}', sep = ""), ]

#######################################################################
# Combine the scndprt variable into 1 row per refdesc + key combination
#######################################################################
part05 <- part04 [, .(scndprt02 = paste(scndprt, collapse = ",", sep = " " )),
                  by = .(refcode, refdesc, frstprt)]
part05 <- part05 [ order(refcode, refdesc, frstprt)]
part05 <- part05 [, rowrecal := .I, by = .(refcode, refdesc)]
```

```
part05 <- part05 [, scndprt03 := paste('"pages": [', scndprt02, "]},", sep=""), ]

chk02 <- part05 [ refcode == "A2.0"]

fwrite(chk02 [ scndprt02 != "...", c("frstprt", "scndprt03"), ],
       "D:/Hospital_data/ProgresSQL/analysis/085d3concept.json",
       col.names = FALSE,
       quote = FALSE,
       sep = " ")

chk02 <- part05 [ refcode == "P5.0"]

fwrite(chk02 [ scndprt02 != "...", c("frstprt", "scndprt03"), ],
       "D:/Hospital_data/ProgresSQL/analysis/085d3concept_P5_0.json",
       col.names = FALSE,
       quote = FALSE,
       sep = " ")

################################################################################
# End of program
################################################################################
```

```
chk02 <- chk02 [, `:=` (name = paste(period, periodn, refcode, refdesc, sep =","),
                        count = cnt,
                        key = paste(Code, description, sep=","),
                        pages = Coded_med,
                        url = Code02,
                        title = Code02),]

chk03 <- chk02 [, c("name", "count", "key", "pages", "url", "title", "nrow"), ]

write_json(chk03,
       "D:/Hospital_data/ProgresSQL/misc/bruce_approach/085d3concept.json")

# Validate Json using https://jsonlint.com/

fwrite(chk01,
       "D:/Hospital_data/ProgresSQL/analysis/085_dis_count_edges_3rd_byPeriod_.csv")


chk01 <- all_met_rmsd02 [, .(cnt = uniqueN(mr_no)),
```

```
                          by = .(refcode, refdesc, Code, description, Type_med, Coded_med
)]
chk01 <- chk01[Code != "" & Coded_med != ""]
chk01 <- chk01 [, Code02 := paste(Code, ":", description, "->", Coded_med, sep =""),]


chk02 <- chk01 [ refcode == "A2.0"]
fwrite(chk02,

"D:/Hospital_data/ProgresSQL/analysis/085_dis_count_edges_3rd_byPeriod_A2_bruce.csv")
################################################################################
# End of program
################################################################################
```

6.5.30 Analysis program for Figure 3-27

```
R program:  080_medicine_repeat_prop.R


library(data.table)
library(stringi)
library(stringr)
library(sqldf)
library(scales)


all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")
```

```
#substr(cat_id, 1, 3) != "SER"
#c("mr_no", "medicine_name", "studyday", "remarks", "frequency", "duration",
"duration_units", "Coded_med", "Type_med",   "quantity", "patient_id", "cat_id")] )

############################
# Data related to medicines
############################
meds0 <- unique( all_met_rmsd [medicine_name != " ",
                               c("mr_no", "studyday", "Coded_med", "Type_med")] )

#############################################################
# Get the minimum day (minday) for any medicine and
# Get the minimum day (minmedday) for individual medicine
#############################################################
meds0 <- meds0 [order(mr_no, studyday)]
meds0 <- meds0 [, minday := min(studyday), by = .(mr_no)]
meds0 <- meds0 [, minmedday := min(studyday), by = .(mr_no, Type_med, Coded_med)]

#############################################################
```

```
# Get group (each day of treatment) as a grouping variable
# Get individual sequential rows within each group
#############################################################
time <- unique(all_met_rmsd [, c("mr_no", "studyday")] )
time <- time [order(mr_no, studyday)]
time <- time [, grpday := 1:.N, by = .(mr_no)]
time <- time [, grpmaxday := max(grpday), by = .(mr_no)]

#########################################################
# Merge the grouping variables for further calculations
# Sort the data
#########################################################
meds0 <- merge (x = meds0, y = time, by = c("mr_no", "studyday") )
meds0 <- meds0[order(mr_no, studyday, grpday)]

#########################################################################
# Sort the data to get prescription number for each medicine
# If the prescription number is > 1 then that medicine is given more than once
#
# There are 2 sequence variables: one for day and one for medicine
```

```
#########################################################################
cum01 <- meds0 [, presc := 1:.N, by = .(mr_no, Type_med, Coded_med)]
cum01 <- cum01 [order(mr_no, studyday, minday, Type_med, Coded_med, presc )]
cum01 <- cum01 [, grpall := 1:.N, by = .(mr_no)]

#################################################################
# If the prescription = 1 and studyday = minmedday then Start
# If prescription > 1 then Old (already given and not a medicine)
# If prescription  group number is > 1 then Start
#################################################################
cum01 <- cum01 [, newold := ifelse (studyday == minmedday, "1st dose", ""), ]
cum02 <- cum01 [, newold2 := ifelse(presc > 1 & grpday > 1 & studyday > minmedday &
newold != "1st dose", "Repeat", newold), by =.(mr_no)]
cum02 <- cum02 [, cat := "Medicine", ]

#########################################################################
# Duplicate the medication and see which medications are given multiple times
# This gives a cumulative view of what has been prescribed till a certain
# Visit, how many medicines are 1st time given and how many are Repeated
#########################################################################
```

```
cum03 <- cum02 [, (list( cumday = (grpday: grpmaxday) ) ),
                by = .(mr_no, presc, Type_med, Coded_med,
                       studyday, grpday, grpmaxday, minmedday, newold2, cat) ]

cum03 <- cum03 [, cumday2 := paste("Till visit", cumday, sep = " "), ]

##############################################################################
# Execute similarly for the diseases area
# check if it is easy to combine disease and medicine like 01_Primary_madhumeha
# display
##############################################################################
#substr(cat_id, 1, 3) != "SER"
# nchar(Code) > 0

############################
# Data related to diseases
############################
meds0 <- unique( all_met_rmsd [, c("mr_no", "Code", "studyday", "description")] )
```

```
meds0 <- data.table(meds0 [, Code := ifelse (Code == " " | Code == "", "** Not yet
coded", Code),])

meds0 <- data.table(meds0 [, description:= ifelse (description == "" | description ==" ",
"** Not yet coded", description),])


############################################################

# Get the minimum day (minday) for any medicine and

# Get the minimum day (minmedday) for individual medicine

############################################################

meds0 <- meds0 [order(mr_no, studyday, Code, description )]

meds0 <- meds0 [, minday := min(studyday), by = .(mr_no)]

meds0 <- meds0 [, minmedday := min(studyday), by = .(mr_no, Code, description)]


#########################################################

# Merge the grouping variables for further calculations

# Sort the data

#########################################################

meds0 <- merge (x = meds0, y = time, by = c("mr_no", "studyday") )

meds0 <- meds0[order(mr_no, studyday, grpday)]


#########################################################################
```

```
# Sort the data to get prescription number for each medicine
# If the prescription number is > 1 then that medicine is given more than once
#
# There are 2 sequence variables: one for day and one for medicine
#########################################################################
cum01 <- meds0 [, presc := 1:.N, by = .(mr_no, Code, description)]
cum01 <- cum01 [order(mr_no, studyday, minday, presc )]
cum01 <- cum01 [, grpall := 1:.N, by = .(mr_no)]

#################################################################
# If the prescription = 1 and studyday = minmedday then Start
# If prescription > 1 then Old (already given and not a medicine)
# If prescription  group number is > 1 then Start
#################################################################
cum01 <- cum01 [, newold := ifelse (studyday == minmedday, "1st time disease", ""), ]
cum02dis <- cum01 [, newold2 := ifelse(presc > 1 & grpday > 1 & studyday > minmedday &
newold != "1st time dose", "Repeat", newold), by =.(mr_no)]
cum02dis <- cum02dis [, cat := "Disease", ]

setnames (cum02dis, "Code", "Type_med")
```

```
setnames (cum02dis, "description", "Coded_med")

#########################################################################
# Duplicate the medication and see which medications are given multiple times
# This gives a cumulative view of what has been prescribed till a certain
# Visit, how many medicines are 1st time given and how many are Repeated
#########################################################################
cum03dis <- cum02dis [, (list( cumday = (grpday: grpmaxday) ) ),
                     by = .(mr_no, Type_med, presc, Coded_med,
                            studyday, grpday, grpmaxday, minmedday, newold2, cat) ]

cum03dis <- cum03dis [, cumday2 := paste("Till visit", cumday, sep = " "), ]

#########################################################
# Combine all disease and medicine information
# for individual visits as well as cumulative visit data
#########################################################
cum02all <- rbind (cum02, cum02dis, fill = TRUE)
cum02all <- cum02all[, -c("newold"),]
cum03all <- rbind (cum03, cum03dis, fill = TRUE)
```

```
fwrite(cum02all,
       "D:/Hospital_data/ProgresSQL/analysis/080_medicine_dis_repeat_prop.csv")
fwrite(cum03all,

"D:/Hospital_data/ProgresSQL/analysis/080_medicine_dis_repeat_prop_cumulative.csv")



all_met_rmsd0 <- data.table(all_met_rmsd [, Code := ifelse (Code == " " | Code == "", "**
Not yet coded", Code),])
all_met_rmsd0 <- data.table(all_met_rmsd0 [, description:= ifelse (description == "" |
description ==" ", "** Not yet coded", description),])
keep <- c("mr_no", "studyday", "patient_gender", "baseage", "age", "Code", "description",
          "Coded_med", "Type_med", "combine", "Metabolic", "RMSD", "vis", "season",
"newdt0", "distype")


all_met_rmsd_unq <- unique( all_met_rmsd0 [, ..keep, ])
all_met_rmsd_unq02 <- merge(x = all_met_rmsd_unq,
                            y = cum02 [, -c("newold", "cat"),],
                            by = c("mr_no", "studyday", "Coded_med", "Type_med"),
                            all.x = TRUE)
```

```
##################################################
# Should look at this syntax for these 2 variables
##################################################
setnames (cum02dis, "Type_med", "Code")
setnames (cum02dis, "Coded_med", "description")

setnames (cum02dis, "presc", "prescdis")
setnames (cum02dis, "newold2", "newold2dis")
setnames (cum02dis, "grpday", "grpdaydis")

all_met_rmsd_unq03 <- merge(x = all_met_rmsd_unq02,
                            y = cum02dis [, -c("newold", "cat", "grpall", "minday",
"minmedday", "grpmaxday"),],
                            by = c("mr_no", "studyday", "Code", "description"),
                            all.x = TRUE)

fwrite(all_met_rmsd_unq03,
       "D:/Hospital_data/ProgresSQL/analysis/080_medicine_dis_all_met_rmsd_prop.csv")
```

```
all_met_rmsd_unq04 <- all_met_rmsd_unq03 [grpday > 0, `:=`(cumday = grpmaxday,
                                                           cumday3 = max(studyday),
                                                           cumday2 = paste("Till visit",
grpmaxday, sep = " ") ),
                                          by = .(mr_no)]
##########################################################################
# Duplicate the medication and see which medications are given multiple times
# This gives a cumulative view of what has been prescribed till a certain
# Visit, how many medicines are 1st time given and how many are Repeated
##########################################################################

all_met_rmsd_unq05 <- all_met_rmsd_unq03 [grpday > 0, (list( cumday = (grpday: grpmaxday)
) ),
                                          by = .(mr_no, presc, prescdis, Type_med,
Coded_med,
                                                 Code, description, baseage, age,
combine, Metabolic, RMSD,
                                                 studyday, grpday, grpmaxday, minmedday,
newold2, newold2dis) ]

all_met_rmsd_unq05 <- all_met_rmsd_unq05 [, cumday2 := paste("Till visit", cumday, sep =
" "), ]
```

```
all_met_rmsd_unq05 <- all_met_rmsd_unq05 [, cumday3 := max(studyday), by = .(mr_no,
cumday2)]



########################################
# Count number of 1st and repeat diseases
# for individual patient
#
# Count number of 1st and repeat doses
# for individual patient
#
# Transpose the
########################################

a0dis <- all_met_rmsd_unq04 [, .(cntdis = uniqueN( paste(Code, description, sep=" "))),
                             by = .(mr_no, grpday, cumday, cumday2, cumday3, newold2dis)]
a0dis_t <- dcast(data = a0dis,
                 mr_no + grpday + cumday + cumday2 + cumday3 ~ newold2dis,
                 value.var = c("cntdis"),
                 fill = 0)
```

```
setnames(a0dis_t, "Repeat", "Repeatdis")


a0dose <- all_met_rmsd_unq04 [, .(cntdose = uniqueN( paste(Type_med, Coded_med, sep="
"))),
                                by = .(mr_no, grpday, cumday, cumday2, cumday3, newold2)]
a0dose_t <- dcast(data = a0dose,
                  mr_no + grpday + cumday + cumday2 + cumday3 ~ newold2,
                  value.var = c("cntdose"),
                  fill = 0)
setnames(a0dose_t, "Repeat", "Repeatdose")


#################################################################
# Count total number of diseases and doses for individual patients
#################################################################
a0distot <- all_met_rmsd_unq05 [, .(totdis = uniqueN( paste(Code, description, sep="
"))),
                                  by = .(mr_no, cumday, cumday2, cumday3)]


a0dosetot <- all_met_rmsd_unq05 [, .(totdose = uniqueN( paste(Type_med, Coded_med, sep="
"))),
                                   by = .(mr_no, cumday, cumday2, cumday3)]
```

```
a01small <- Reduce(function(...) merge(..., all.y = TRUE, by = c("mr_no", "grpday",
"cumday", "cumday2", "cumday3") ),
                   list(a0dis_t, a0dose_t))


a01cap <- Reduce(function(...) merge(..., all.y = TRUE, by = c("mr_no", "cumday",
"cumday2", "cumday3") ),
                  list(a0distot, a0dosetot))
a01all <- merge(x = a01small [, -c("cumday", "cumday2", "cumday3"),],
                y = a01cap,
                by.x = c("mr_no", "grpday"),
                by.y = c("mr_no", "cumday"))
a01all <- a01all [, `:=` (perc1dis = percent(`1st time disease` / totdis),
                          percrepdis = percent(`Repeatdis` / totdis),
                          perc1dose = percent(`1st dose` / totdose),
                          percrepdose = percent(`Repeatdose` / totdose)) , ]


################################################
# Get the diseases and doses collapsed into 1 row
################################################
```

```
dis <- unique(all_met_rmsd_unq03 [grpday > 0,
                                  c("mr_no", "grpday", "Code", "description", "distype",
"studyday", "newold2dis"), ])
dis <- dis [, `:=` (disall = paste(distype, Code, sep= ":"),
                    desall = paste(distype, description, sep= ":"))]

discomb <- dis [grpday > 0,
                .(discomb = paste(disall, collapse = ";", sep = " " ),
                     descomb = paste(desall, collapse = ";", sep = " " )),
                by = .(mr_no, grpday, newold2dis)]
discomb_t <- dcast(data = discomb,
                   mr_no + grpday ~ newold2dis,
                   value.var = c("discomb", "descomb"))

dose <- unique(all_met_rmsd_unq03 [grpday > 0,
                                  c("mr_no", "grpday", "Type_med", "Coded_med",
"studyday", "newold2"), ])
dose <- dose [, doseall := paste(Type_med, Coded_med, sep= ":")]
doscomb <- dose [grpday > 0,
                .(dosecomb = paste(doseall, collapse = ";", sep = " " )),
                by = .(mr_no, grpday, newold2)]
```

```
doscomb_t <- dcast(data = doscomb,
                   mr_no + grpday ~ trimws(paste("Combine", newold2, sep="")),
                   value.var = c("dosecomb"))


adsl <- unique( all_met_rmsd_unq03 [grpday >0, c("mr_no", "patient_gender", "grpday",
"season",
                                        "age", "baseage", "combine", "Metabolic",
"RMSD"), ])


a01all <- Reduce(function(...) merge(..., all.y = TRUE, by = c("mr_no", "grpday") ),
                   list(a01all, discomb_t, doscomb_t))


a01all <- merge(x = a01all,
                y = adsl,
                by = c("mr_no", "grpday"))
fwrite(a01all,

"D:/Hospital_data/ProgresSQL/analysis/080_medicine_repeat_prop_cumulative_Rcal.csv")
saveRDS (a01all,
"D:/Hospital_data/ProgresSQL/analysis/080_medicine_repeat_prop_cumulative_Rcal.rds")
```

```
########################################
# Create disease and medicine combination
# by patient
########################################
dismed <- all_met_rmsd_unq04 [, .(cntdismed = .N),
                              by = .(mr_no,
                                     Code, description, Type_med, Coded_med)]
dis100 <- all_met_rmsd_unq04 [, .(cntdis = .N),
                              by = .(mr_no,
                                     Code, description)]
med100 <- all_met_rmsd_unq04 [, .(cntmed = .N),
                              by = .(mr_no,
                                     Type_med, Coded_med)]

dismed01 <- merge(x = dismed,
                  y = dis100,
                  by = c("mr_no", "Code", "description"),
                  all = TRUE)
dismed02 <- merge(x = dismed01,
                  y = med100,
```

```
                  by = c("mr_no", "Type_med", "Coded_med"),
                  all = TRUE)
fwrite(dismed02,
       "D:/Hospital_data/ProgresSQL/analysis/080_medicine_bymr_no_dismed_comb_Rcal.csv")
saveRDS (dismed02,
"D:/Hospital_data/ProgresSQL/analysis/080_medicine_bymr_no_dismed_comb_Rcal.rds")

##########################################
# Create disease and medicine combination
# by medicine and disease
# count number of combinations and number of
# patients
##########################################
a_dismed <- all_met_rmsd_unq04 [, .(cntdismed = .N,
                                    unqdismedpat = uniqueN(mr_no)),
                                 by = .(Code, description, Type_med, Coded_med)]

a_dis100 <- all_met_rmsd_unq04 [, .(cntdis = .N, unqdispat = uniqueN(mr_no)),
                                 by = .(Code, description)]
```

```
a_med100 <- all_met_rmsd_unq04 [, .(cntmed = .N, unqmedpat = uniqueN(mr_no)),
                                by = .(Type_med, Coded_med)]

a_dismed01 <- merge(x = a_dismed,
                    y = a_dis100,
                    by = c("Code", "description"),
                    all = TRUE)

a_dismed02 <- merge(x = a_dismed01,
                    y = a_med100,
                    by = c("Type_med", "Coded_med"),
                    all = TRUE)

fwrite(a_dismed02,

"D:/Hospital_data/ProgresSQL/analysis/080_medicine_byoverall_dismed_comb_Rcal.csv")

saveRDS (a_dismed02,
"D:/Hospital_data/ProgresSQL/analysis/080_medicine_byoverall_dismed_comb_Rcal.rds")

################################################################################
# End of program
################################################################################
```

6.5.31 Analysis program for Figure 3-28

Refer to R program: 080_medicine_repeat_prop.R

6.5.32 Analysis program for Figure 3-29

Refer to R program: 080_medicine_repeat_prop.R

6.5.33 Analysis program for Figure 3-30

R program: diagnosis.R

```
#################################

# Code to generate DIAGNOSIS data

#################################


library(data.table)

setwd ("C:\\Users\\Lucky\\Documents\\Hospital data\\01_31JUL2016\\source")


diag <- fread("Diagnosis.csv", check.names = FALSE)


diag <- diag[, `:=` (data = "diag",
                     type = substr(`Patient Id`, 1, 2) ), ]


diag2 <- diag[, c(1, 4, 5, 6, 8, 9, 10, 23, 29, 34, 44, 118, 119), with=FALSE ]
```

```
setnames(diag2, "MR No.", "MRNo")
setnames(diag2, "Admission Date", "visdate")
setnames(diag2, "Blood Group", "Bloodgrp")
setnames(diag2, "Age In", "AgeIn")
setnames(diag2, "First Visit Date", "fvisdate")

#setnames(diag2, "Diagnosis Type", "diagtype")
# Create date variables and find the difference
diag2 <- diag2[, visdate := as.POSIXct( gsub("-", "/", visdate), format="%d/%m/%Y") ]
diag2 <- diag2[, fvisdate := as.POSIXct( gsub("-", "/", fvisdate), format="%d/%m/%Y") ]
diag2 <- diag2[, visday := as.Date(visdate) - as.Date(fvisdate) + 1]
diag2 <- diag2[, vismon := round(visday /30.4375, 1)]
diag2 <- diag2[, Age := ifelse(AgeIn =="M", Age/12, Age), ]
diag2 <- diag2[, AgeIn := ifelse(AgeIn =="M", "Y", AgeIn), ]

# Code ** NOT YET CODED for the missing code values
# Sort the data by patient and day
diag2 <- diag2[, Code2 := ifelse(Code =="", "ZZZ999", Code), ] [order(MRNo, visday,
Code2)]
```

```
# Create number of diagnosis per patient and a counter for each diagnosis
diag2 <- diag2[ , noofdis := uniqueN(Code2), by = .(MRNo) ]

# Count number of unique diagnosis per patient per day
# Sort the data by patient and day
diag2 <- diag2[ , `:=` (IDX = 1: .N), by = .(MRNo, visdate, visday) ] [order(MRNo,
visday, IDX, Code2)]

# Get the first date of the diagnosis by each code
diag2 <- diag2 [, min := min(visday), by = .(MRNo, Code2)]

# Get the maximum date of the diagnosis (end date for each patient)
# Use this data with vital sign data to get diagnosis attached to vital sign measurements
diag2 <- diag2 [, max := max(visday), by = MRNo]

setkeyv (diag2, c("MRNo", "Code2", "Description", "min", "max", "type", "IDX",
"noofdis"))

diag2 <- unique(diag2)
write.csv(diag2, file ="C:\\Users\\mahajvi1\\Desktop\\adiag.csv", na=" ")
##########################################################################
```

```
# End of program
#########################################################################################
```

6.5.34 Analysis program for Figure 3-31

```
R program:  305_medicine_duration_by_dis.R


#########################################################################
# Medicine duration
# Medicine duration by disease
# Summary statistics should show the most frequently used medicines
# Indirect relationship builiding
# More usage stronger the relationship
# Less usage may be no relationship or rare usage
#########################################################################

library(data.table)
library(tidyverse)
library(sqldf)

all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")
all_met_rmsd <- all_met_rmsd [, Code := ifelse (Code == " " | Code == "", "** Not yet
coded", Code),]
```

```
all_met_rmsd <- all_met_rmsd [, description:= ifelse (description == "" | description ==" ", "** Not yet coded", description),]

all_met_rmsd <- all_met_rmsd [, Code02 := paste(distype, ":", Code, ":", description, sep =""), ]

all_met_rmsd <- all_met_rmsd [, Med02 := paste(Type_med, ":", Coded_med, sep =""), ]


med01 <- unique( all_met_rmsd [Med02 != "NA:NA" ,
                               c("mr_no", "Med02", "Type_med", "Coded_med", "studyday",
                                 "frequency", "duration", "duration_units", "cat_id",
                                 "patient_gender"),])

med01 <- med01 [ duration > 0]

med01 <- med01 [, duration := as.numeric(duration),]

med01 <- med01 [, numdays := case_when( duration_units == "D" ~ duration,
                                        duration_units == "W" ~ duration * 7,
                                        duration_units == "M" ~ duration * 30), ]


# Create 1 record per patient per medication with sum of durations

med011 <- med01 [, .(numdays = sum(numdays)), by =.(mr_no, Med02, Type_med, Coded_med, patient_gender)]

med02 <- med011 [, .(n=uniqueN(mr_no),
                     mean = round( mean(numdays, na.rm = TRUE), digits =1),
```

```
                    median= round( median(numdays, na.rm = TRUE), digits =2),
                    SD = round( sd(numdays, na.rm = TRUE), digits =2),
                    min = round( min(numdays, na.rm = TRUE), digits =0),
                    max = round( max(numdays, na.rm = TRUE), digits =0),
                    sum = round( sum(numdays, na.rm = TRUE), digits =0)),
                by = .(Type_med)]

med02_med <- med011 [, .(n=uniqueN(mr_no),
                        mean = round( mean(numdays, na.rm = TRUE), digits =1),
                        median= round( median(numdays, na.rm = TRUE), digits =2),
                        SD = round( sd(numdays, na.rm = TRUE), digits =2),
                        min = round( min(numdays, na.rm = TRUE), digits =0),
                        max = round( max(numdays, na.rm = TRUE), digits =0),
                        sum = round( sum(numdays, na.rm = TRUE), digits =0)),
                    by = .(Type_med, Med02)]

dismed01 <- unique( all_met_rmsd [Med02 != "NA:NA" ,
                                  c("mr_no", "Med02", "Type_med", "Coded_med",
"studyday",
                                    "Code02",
```

```
                                    "frequency", "duration", "duration_units", "cat_id",
                                    "patient_gender"),])
dismed01 <- dismed01 [ duration > 0]
dismed01 <- dismed01 [, duration := as.numeric(duration),]
dismed01 <- dismed01 [, numdays := case_when( duration_units == "D" ~ duration,
                                              duration_units == "W" ~ duration * 7,
                                              duration_units == "M" ~ duration * 30), ]

# Create 1 record per patient per medication with sum of durations
dismed011 <- dismed01 [, .(numdays = sum(numdays)), by =.(mr_no, Code02, Med02, Type_med,
Coded_med, patient_gender)]

# Create count of patients with
# totpatdis: total patients having the disease
# totpatmed: total patients having the medicine prescribed

dismed011 <- dismed01 [, totpatdis := uniqueN(mr_no), by =.(Code02)]
dismed011 <- dismed01 [, totpatmed := uniqueN(mr_no), by =.(Med02)]
dismed02 <- dismed011 [, .(n=uniqueN(mr_no),
                           mean = round( mean(numdays, na.rm = TRUE), digits =1),
```

```
                            median= round( median(numdays, na.rm = TRUE), digits =2),
                            SD = round( sd(numdays, na.rm = TRUE), digits =2),
                            min = round( min(numdays, na.rm = TRUE), digits =0),
                            max = round( max(numdays, na.rm = TRUE), digits =0),
                            sum = round( sum(numdays, na.rm = TRUE), digits =0)),
                          by = .(Type_med, Code02, totpatdis, totpatmed)]

dismed02_med <- dismed011 [, .(n=uniqueN(mr_no),
                               mean = round( mean(numdays, na.rm = TRUE), digits =1),
                               median= round( median(numdays, na.rm = TRUE), digits =2),
                               SD = round( sd(numdays, na.rm = TRUE), digits =2),
                               min = round( min(numdays, na.rm = TRUE), digits =0),
                               max = round( max(numdays, na.rm = TRUE), digits =0),
                               sum = round( sum(numdays, na.rm = TRUE), digits =0)),
                            by = .(Code02, totpatdis, Type_med, Med02, totpatmed)]

# n: total number of patients having the disease and medicine prescribed
# So n and totpatmed: calculate the %
dismed02_med <- dismed02_med [, perc := round ( n / totpatmed * 100, digits = 2),]
##############################################################################
```

```
# End of program
#############################################################################
```

6.5.35 Analysis program for Figure 3-32

```
Refer to R program:  305_medicine_duration_by_dis.R
```

6.5.36 Analysis program for Figure 3-33

```
Refer to R program:  305_medicine_duration_by_dis.R
```

6.5.37 Analysis program for Figure 3-34

```
Refer to R program:  305_medicine_duration_by_dis.R
```

6.5.38 Analysis program for Figure 3-35

```
Refer to R program:  100_adsl.R
```

6.5.39 Analysis program for Figure 3-36

```
Refer to R program:  100_adsl.R
```

6.5.40 Analysis program for Figure 3-37

```
R program:   085_dis_counts_edges_3rdbyPeriod.R


#################################################################

# This is used for 085_dis_count_edges_3rd_byPeriod Tableau display

#################################################################


library(data.table)

library(stringi)

library(stringr)
```

```
library(sqldf)
library(tidyr)

#####################################################################
# These 2 are created using 085_dis_1st_time_refCal_NodeEdges.R program
#####################################################################

all_met_rmsd02 <- readRDS ("D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")
all_met_rmsd02 <- all_met_rmsd02 [, Coded_med := paste(Type_med, Coded_med, sep=":"),]

edges <-
readRDS("D:/Hospital_data/ProgresSQL/analysis/085_dis_1st_time_refCal_NodesEdges.rds")
#####################################################
# Get unique diseases in RMSD and Metabolic
# Keep refcode from these 2 areas 107 unique values
#####################################################
discat <- unique( all_met_rmsd02 [distype %in% c("RMSD", "Metabolic"), c("Code",
"description"), ])

#########################################################
# Get the unique number of reference diseases and medicines
```

```
# Create this for each of the periods before and after 1st
# occurrence of the disease
#########################################################
unqref <- unique( edges [, c("period", "periodn")])

dismed <- unique( edges [, c("cat", "refcode", "refdesc", "Code", "description"), ])
dismed <- dismed [nchar(Code) > 0] [order(cat, refcode, refdesc, Code, description)]
dismed <- dismed [, `:=` (npoints = 1:.N, tot = .N), by = .(cat, refcode, refdesc)]
dismed <- dismed [, `:=` (radius = ifelse (cat == "Disease", 20, 40),
                          angle = 360 / tot), ]
dismed <- dismed [, cumulative := cumsum(angle), by = .(cat, refcode, refdesc)]

###############################################
# Function for the degrees and radian conversion
###############################################
deg2rad <- function(deg) {(deg * pi) / (180)}

dismed <- dismed [, radian := deg2rad(cumulative),]
dismed <- dismed [, `:=` (xaxis = cos(radian)*radius,
                          yaxis = sin(radian)*radius), ]
```

```
#dismed <- dismed [, Code02 := paste(Code, ":", description), ]
#dismed <- dismed [, cnt := 0,]

#########################################################
# create a complete dataset
# this is to ensure, circle is displayed all the time
# Combine with the individual period for replication
#########################################################
dismed_all <- crossing(unqref, dismed)

chk01 <- all_met_rmsd02 [, .(cnt = uniqueN(mr_no)),
                         by = .(period, periodn, refcode, refdesc, Code, description,
Type_med, Coded_med )]
chk01 <- chk01[Code != "" & Coded_med != ""]
chk01 <- chk01 [, Code02 := paste(Code, ":", description, "->", Coded_med, sep =""),]

# Merge the x and y coordiantes

chk02dis <- unique(chk01 [, c("period","periodn", "refcode", "refdesc", "Code",
"description", "cnt", "Code02")] )
```

```
chk02dis <- chk02dis [, cat := "Disease"]


chk02med <- unique(chk01 [, c("period","periodn", "refcode", "refdesc", "Type_med",
"Coded_med", "cnt", "Code02")])
setnames (chk02med, "Type_med", "Code")
setnames (chk02med, "Coded_med", "description")
chk02med <- chk02med [, cat := "Medicine"]


chk02all <- rbind(chk02dis, chk02med)
chk02all <- chk02all [nchar(Code) >0 & nchar(description) > 0 ]


path01 <- merge (x = chk02all,
                 y = dismed_all,
                 by = c("cat", "period","periodn", "refcode", "refdesc", "Code",
"description"),
                 all = TRUE)
path01 <- path01 [, `:=` (cat = "DiseaseMedicine", Code = Code02),]
path01 <- path01 [, c("TabCode", "TabMed") := tstrsplit(Code, "->", fixed = TRUE), ]


chk03all <- rbind(path01, dismed_all, fill = TRUE)
chk04all <- chk03all [ refcode %in% discat$Code]
```

```
fwrite(chk04all,
       "D:/Hospital_data/ProgresSQL/analysis/085_dis_count_edges_3rd_byPeriod.csv")
#############################################################################
```

6.5.41 Analysis program for Figure 3-38

```
Refer to R program: 085_dis_counts_edges_3rdbyPeriod.R
```

6.5.42 Analysis program for Figure 3-39

```
R program:    086time_dis_patterns_combinations_gender_Macro.R


#####################################################################
# 086time_dis_patterns_combinations_gender_Macro.R
#####################################################################

library(tidyverse)
library(tidytext)
#library(stringr)
library(stringi)
library(data.table)
library(stringdist)
library(scales)
```

```
# https://stackoverflow.com/questions/43706729/expand-dates-in-data-table
dis <- fread("D:/Hospital_data/ProgresSQL/analysis/discategory.csv")
setnames (dis, "Code", "refcode")

all_met_rmsd <- readRDS ("D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")

##############################################
# Find patients with only the disease
# same as reference disease
# 1 = patients with only disease
# 99 = patients with more than 1 disease in
# a reference disease category
##############################################

addmr <- unique( all_met_rmsd [!Code %in% c(" ", ""), c("mr_no", "refcode", "Code",
"distype"),])
addmr <- addmr [, cnt := uniqueN(refcode), by  = .(mr_no)]
addmr <- addmr [, dis := ifelse(refcode == Code, 1, 0),]
addmr <- addmr [, calc := ifelse(cnt == 1 & dis == 1, 1, 99),]
```

```
addmr02 <- addmr [, .(cntr = uniqueN(mr_no)), by = .(distype, refcode, Code, calc)]
addmr03 <- addmr [, .(cntot = uniqueN(mr_no)), by = .(refcode)]
addmr04 <- merge(addmr02, addmr03, by = c("refcode"))
addmr04 <- addmr04 [, perc := percent(cntr / cntot),]
addmr05 <- addmr04 [ refcode == Code]
addmr06 <- merge (addmr05, dis, by = c("refcode"), all.y = TRUE)

unq <- unique(addmr06 [cntot > 5, c("refcode"),])
unqdis <- unique(unq$refcode)

count <- 1

for ( dis in unqdis[1:uniqueN(unqdis)])
{ print (dis)
  print (count)

  a2 <- all_met_rmsd [!Code %in% c("", " ", dis) & refcode == dis]
  a2 <- a2 [, Code := paste(period, Code, sep="_"),]
```

```
  #a2med <- a2 [, description := paste(Type_med, Coded_med),]

  #a2med <- a2med [ order(description)]

  #a2med <- a2med [, Code := paste("M", str_pad(.N, width =4, pad="0"), sep =""), by =
.(description) ]


  #a2all <- rbind(a2 [, c("mr_no", "studyday", "refday", "Code", "description",
"refcode", "refdesc", "patient_gender")],

  #               a2med [, c("mr_no", "studyday", "refday", "Code", "description",
"refcode", "refdesc", "patient_gender")] )


  # Change a2 to a2all

  dis <- unique(a2[, c("mr_no", "studyday", "refday", "Code", "description", "refcode",
"refdesc", "patient_gender")])

  dis <- dis [, `:=` (refday2 = ifelse(refday >=1, "After", "Before"),

                      Code = str_replace_all (Code, " ", ""),

                      description = str_replace_all(description, " ", "")),]

  dis <- dis [ order(mr_no, studyday, Code, refcode, refdesc)]

  dis <- dis [, `:=` (alldis = uniqueN(Code),

                      nrow = seq_len(.N),

                      nrowend = seq_len(.N) + 4,

                      totrow = .N), by = .(mr_no, refcode, refdesc)]
```

```
  dis <- dis [, `:=` (alldisbfraftr = uniqueN(Code),
                      nrowbfraftr = seq_len(.N) ), by = .(mr_no, refcode, refdesc,
refday2, patient_gender)]
  dis <- dis [, `:=` (total = uniqueN(mr_no) ), by = .(refcode, refdesc, refday2,
patient_gender)]
  dis <- dis [, `:=` (allcapn = uniqueN(mr_no) ), by = .(refcode, refdesc,
patient_gender)]


  dis02 <- dis [, .(combdis = paste(unique(Code), collapse = ",", sep = " " ),
                    combdesc = paste(unique(description), collapse = ",", sep = " " )),
                by = .(mr_no, refcode, refdesc, refday2, patient_gender, alldis, totrow,
total, allcapn)]


  unq01comb <- unique( dis02 [, c("mr_no", "refcode", "refdesc", "alldis",
"refday2","patient_gender",
                                  "totrow",  "combdis", "combdesc", "total", "allcapn"),
])
  unq01comb <- unq01comb [, x := 1, ]


  # create a copy
  unq02comb <- copy(unq01comb)
  setnames(unq02comb, "mr_no", "mr_no2")
```

```
  setnames(unq02comb, "combdis", "combdis2")

  unq01comb <- unq01comb [, combdis := str_replace_all(combdis, ",", "|"), ]

  # Merge the datasets on x to get all the combinations

  unq03comb <- merge(x = unq01comb,
                     y = unq02comb [, -c("refcode", "refdesc", "totrow", "alldis",
"total", "allcapn", "combdesc"), ],
                     by = c("x", "refday2", "patient_gender"),
                     allow.cartesian = TRUE)

  ##########################################################
  # Using str_count function to count the common diseases
  # Create tempdis and tempdis2
  #
  # Consider mr_no as the reference patient
  # tempdis: should be lookup
  # a: common in both the strings
  # b: only present in reference patient (mr_no)
```

```
# c: only present in other patient (mr_no2)
# d: complete absence -- not sure how to calculate this
##########################################################

unq03comb <- unq03comb [, `:=` (tempdis = str_replace_all(combdis, ",", "|"),
                                tempdis2 = str_replace_all(combdis2, ",", "|")),]

unq03comb <- unq03comb [, `:=` (cntdis = str_count(tempdis, "\\|") + 1,
                                cntdis2 = str_count(tempdis2, "\\|") + 1), ]

unq03comb <- unq03comb[, `:=` (a = str_count(combdis2, tempdis)),]

unq03comb <- unq03comb [, `:=` (b = cntdis - a,
                                c = cntdis2 - a),  ]

unq03comb <- unq03comb[, `:=` (a01jac = (a / (a + b + c)),
                               a02dice = (2 * a / (2* a + b + c) ),
                               a03CZEKANOWSKI = (2 * a / (2* a + b + c) ),
                               a04jac3w = (3 * a / (3* a + b + c) ),
                               a05nei_li = (2 * a /  (a + b + a + c) ),
```

```
                                    a06sokalsneath1 = (a / (a + 2 * b + 2 * c)) ),]
  unq03comb <- unq03comb [ mr_no != mr_no2]


  maxscr <- unq03comb[, .(maxscr = max(a01jac) ), by = .(mr_no, refcode, total, allcapn,
totrow, alldis, refday2, patient_gender, combdis, combdesc)]
  maxscr_t <- dcast (data = maxscr,
                     mr_no + patient_gender + refcode + totrow + alldis ~ refday2,
                     value.var = c("maxscr", "combdis", "combdesc"))


  maxscr02 <- maxscr [, .(scr = uniqueN(mr_no)), by = .(refcode, total, allcapn, refday2,
patient_gender, cut(maxscr,

seq(0, 1, .25),

include.lowest = TRUE,

ordered_result = TRUE))]
  maxscr02_t <- dcast(data = maxscr02,
                      refcode + patient_gender + allcapn + cut ~ refday2,
                      value.var = c("scr", "total"))
```

```
  maxscr03 <- maxscr [, .(scr = uniqueN(mr_no)), by = .(refcode, total, allcapn, maxscr,
combdis, combdesc, refday2, patient_gender)]
  maxscr03_t <- dcast(data = maxscr03,
                      refcode + patient_gender + allcapn + combdis + combdesc + maxscr ~
refday2,
                      value.var = c("scr", "total"))


  maxscr04_t <- unq03comb [, .(scr = .N), by = .(mr_no, refcode, total, allcapn, refday2,
patient_gender, cut(a01jac,

seq(0, 1, .25),

include.lowest = TRUE,

ordered_result = TRUE))]
  maxscr04_t <- maxscr04_t [, numrow := .N, by = . (mr_no, refcode, refday2,
patient_gender)]


  totscr <- unq03comb [, .( rowcnt = .N), by = .(refcode, total, allcapn, refday2,
patient_gender, cut(a01jac,

seq(0, 1, .25),

include.lowest = TRUE,
```

```
ordered_result = TRUE) )]
  totscr02 <- unq03comb [, .(totn = .N), by = .(refcode, refday2, patient_gender, total,
allcapn )]
  totscr02 <- merge (totscr, totscr02, by = c("refcode", "refday2", "patient_gender",
"total", "allcapn"))
  totscr02 <- totscr02 [, perc := percent( rowcnt / totn),]
  totscr02_t <- dcast(data = totscr02,
                      refcode + patient_gender + allcapn + cut ~ refday2,
                      value.var = c("perc", "totn", "rowcnt", "total"))

  assign ( paste("D01maxscr_t", count, sep="") ,  maxscr_t)
  assign ( paste("D02maxscr02_t", count, sep="") ,  maxscr02_t)
  assign ( paste("D03maxscr03_t", count, sep="") ,  maxscr03_t)
  assign ( paste("D03maxscr04_t", count, sep="") ,  maxscr04_t)
  assign ( paste("t02totscr02_t", count, sep="") ,  totscr02_t)

  count = count + 1

}
```

```
allD01maxscr_t <- rbindlist(mget(ls(pattern = "D01maxscr_t*")), fill = TRUE)
allD02maxscr02_t <- rbindlist(mget(ls(pattern = "D02maxscr02_t*")), fill = TRUE)
allD03maxscr03_t <- rbindlist(mget(ls(pattern = "D03maxscr03_t*")), fill = TRUE)
allD03maxscr04_t <- rbindlist(mget(ls(pattern = "D03maxscr04_t*")), fill = TRUE)

allt02maxscr02_t <- rbindlist(mget(ls(pattern = "t02totscr02_t*")), fill = TRUE)

rm(list = ls( pattern='^D01maxscr_t*'))
rm(list = ls( pattern='^D02maxscr02_t*'))
rm(list = ls( pattern='^D03maxscr03_t*'))
rm(list = ls( pattern='^D03maxscr04_t*'))
rm(list = ls( pattern='^t02totscr02_t*'))

fwrite(allD01maxscr_t, "D:/Hospital_data/ProgresSQL/analysis/086time_dis_indPat_max.csv")
fwrite(allD02maxscr02_t,
"D:/Hospital_data/ProgresSQL/analysis/086time_dis_refcode_max.csv")
fwrite(allD03maxscr03_t,
"D:/Hospital_data/ProgresSQL/analysis/086time_dis_indtrajectory.csv")
fwrite(allD03maxscr04_t,
"D:/Hospital_data/ProgresSQL/analysis/086time_dis_indPat_freqcat.csv")
```

```
fwrite(allt02maxscr02_t,
"D:/Hospital_data/ProgresSQL/analysis/086time_dis_refcode_allfreq_max.csv")
################################################################################
# End of program
################################################################################
```

6.5.43 Analysis program for Figure 3-40

R program: 086_med_patterns_combinations.R

```
#################################################################

# 086_med_patterns_combinations

#################################################################


library(tidyverse)

library(tidytext)

library(stringr)

library(stringi)

library(data.table)

library(stringdist)


all_met_rmsd <- readRDS ("D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")

a2 <- all_met_rmsd [ refcode == "A2.0" & Coded_med != " "]
```

```
a2 <- a2 [, description := paste(Type_med, Coded_med),]

a2 <- a2 [ order(description)]

a2 <- a2 [, Code := paste("M", str_pad(.N, width =4, pad="0"), sep =""), by =
.(description) ]


dis <- unique(a2[!Code %in% c("", " ", "A2.0") & refcode == "A2.0", c("mr_no",
"studyday", "refday", "Code", "description", "refcode", "refdesc")])

dis <- dis [, `:=` (refday2 = ifelse(refday >=1, "After", "Before"),

                    Code = str_replace_all (Code, " ", ""),

                    description = str_replace_all(description, " ", "")),]

dis <- dis [ order(mr_no, studyday, Code, refcode, refdesc)]

dis <- dis [, `:=` (alldis = uniqueN(Code),

                    nrow = seq_len(.N),

                    nrowend = seq_len(.N) + 4,

                    totrow = .N), by = .(mr_no, refcode, refdesc)]

dis <- dis [, `:=` (alldisbfraftr = uniqueN(Code),

                    nrowbfraftr = seq_len(.N) ), by = .(mr_no, refcode, refdesc,
refday2)]


dis02 <- dis [, .(combdis = paste(unique(Code), collapse = ",", sep = " " )),

              by = .(mr_no, refcode, refdesc, refday2, alldis, totrow)]
```

```
unq01comb <- unique( dis02 [, c("mr_no", "refcode", "refdesc", "alldis", "refday2",
                                "totrow",  "combdis"), ])
unq01comb <- unq01comb [, x := 1, ]

# create a copy
unq02comb <- copy(unq01comb)
setnames(unq02comb, "mr_no", "mr_no2")
setnames(unq02comb, "combdis", "combdis2")

unq01comb <- unq01comb [, combdis := str_replace_all(combdis, ",", "|"), ]

# Merge the datasets on x to get all the combinations

unq03comb <- merge(x = unq01comb,
                   y = unq02comb [, -c("refcode", "refdesc", "totrow", "alldis"), ],
                   by = c("x", "refday2"),
                   allow.cartesian = TRUE)

#########################################################
```

```
# Using str_count function to count the common diseases
# Create tempdis and tempdis2
#
# Consider mr_no as the reference patient
# tempdis: should be lookup
# a: common in both the strings
# b: only present in reference patient (mr_no)
# c: only present in other patient (mr_no2)
# d: complete absence -- not sure how to calculate this
#########################################################

unq03comb <- unq03comb [, `:=` (tempdis = str_replace_all(combdis, ",", "|"),
                                tempdis2 = str_replace_all(combdis2, ",", "|")),]
unq03comb <- unq03comb [, `:=` (cntdis = str_count(tempdis, "\\|") + 1,
                                cntdis2 = str_count(tempdis2, "\\|") + 1), ]
unq03comb <- unq03comb[, `:=` (a = str_count(combdis2, tempdis)),]

unq03comb <- unq03comb [, `:=` (b = cntdis - a,
                                c = cntdis2 - a),  ]
```

```
unq03comb <- unq03comb[, a01jac := (a / (a + b + c)),]

distraj <- unique(unq03comb [tempdis != tempdis2 , c("tempdis", "tempdis2", "a01jac",
"refday2"),])

distraj01 <- distraj [, .(distraj = .N), by = .(refday2, a01jac)]

common <- unq03comb [, .(cmn = (.N / nrow(unq03comb)) * 100), by = .(a)]

################################################################################
# End of program
################################################################################
```

6.5.44 Analysis program for Figure 3-41

```
R program: 300_radar_plot_tableu.R


# Tableu help

# Use https://www.tableau.com/about/blog/2015/7/use-radar-charts-compare-dimensions-over-
several-metrics-41592


library(data.table)

library(tidyverse)

library(sqldf)


# Install the ggradar library

#devtools::install_github("ricardo-bion/ggradar", dependencies = TRUE)
```

```
library(ggradar)


all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")

all_met_rmsd <- all_met_rmsd [, Code := ifelse (Code == " " | Code == "", "** Not yet
coded", Code),]

all_met_rmsd <- all_met_rmsd [, description:= ifelse (description == "" | description ==""
", "** Not yet coded", description),]

all_met_rmsd <- all_met_rmsd [, Code02 := paste(distype, ":", Code, ":", description, sep
=""), ]

all_met_rmsd <- all_met_rmsd [, Med02 := paste(Type_med, ":", Coded_med, sep =""), ]


all_met_rmsd02 <- readRDS("D:/Hospital_data/ProgresSQL/analysis/all_met_rmsd02.rds")

all_met_rmsd02 <- all_met_rmsd02 [, Code := ifelse (Code == " " | Code == "", "** Not yet
coded", Code),]

all_met_rmsd02 <- all_met_rmsd02 [, description:= ifelse (description == "" | description
==" ", "** Not yet coded", description),]

all_met_rmsd02 <- all_met_rmsd02 [, Code02 := paste(distype, ":", Code, ":", description,
sep =""), ]

all_met_rmsd02 <- all_met_rmsd02 [, Med02 := paste(Type_med, ":", Coded_med, sep =""), ]


# Disease:

# (1) Distinct number of patients
```

```
t10 <- all_met_rmsd02 [, .(cal10 = uniqueN(mr_no)), by =.(refcode, refdesc)]
t10 <- t10 [, perc10 := as.numeric( ntile(cal10, 100) ), ]

# (2) Number of times a disease is reported
totdis <- unique( all_met_rmsd [ , c("Code", "description", "patient_id"),] )

t20 <- totdis [, .(cal20 = .N), by =.(Code, description)]
t20 <- t20 [, perc20 := as.numeric( ntile(cal20, 100) ), ]

# (3) Number for a specific disease (chronological number of disease reported by a
patient)
# Calculate median number of disease reported for each disease
# Then calculate the percentile for each disease

numdis <- unique( all_met_rmsd [, c("mr_no", "Code", "description", "studyday"),])
numdis <- numdis [ order(mr_no, studyday, Code)]
numdis <- numdis [ , `:=`( COUNT = .N , ndis = 1:.N ),
                by = .(mr_no, Code, description) ]

t30 <- numdis [, .(cal30 = median(ndis)), by =.(Code, description)]
```

```
t30 <- t30 [, perc30 := as.numeric( ntile(cal30, 100) ), ]


# (4) Number of diseases before and after the specific disease
banumdis <- unique( all_met_rmsd02 [, c("mr_no", "Code", "description", "period",
"periodn", "refcode", "refdesc"),])
banumdis <- banumdis [, classification := ifelse (period >=1 , "After", "Before"), ]
banumdis <- banumdis [ order(mr_no, refcode, refdesc, classification)]
banumdis <- banumdis [ , `:=`( ndis = uniqueN(Code) ),
                   by = .(mr_no, refcode, refdesc, classification) ]


t40 <- banumdis [, .(cal40 = median(ndis)), by =.(refcode, refdesc, classification)]
t40 <- t40 [, perc40 := as.numeric( ntile(cal40, 100) ), ]
t40_trn <- dcast(data = t40,
                 refcode + refdesc ~ classification,
                 value.var = c("cal40", "perc40"),
                 fill ="0")


# (5) Number of treatments before and after the specific disease
banummed <- unique( all_met_rmsd02 [, c("mr_no", "Med02", "period", "periodn", "refcode",
"refdesc"),])
banummed <- banummed [, classification := ifelse (period >=1 , "After", "Before"), ]
```

```
banummed <- banummed [ order(mr_no, refcode, refdesc, classification)]
banummed <- banummed [ , `:=`( nmed = uniqueN(Med02)),
                       by = .(mr_no, refcode, refdesc, classification) ]

t50 <- banummed [, .(cal50 = median(nmed)), by =.(refcode, refdesc, classification)]
t50 <- t50 [, perc50 := as.numeric( ntile(cal50, 100) ), ]
t50_trn <- dcast(data = t50,
                 refcode + refdesc ~ classification,
                 value.var = c("cal50", "perc50"),
                 fill ="0")

#setnames(t10, "Code", "refcode")
setnames(t20, "Code", "refcode")
setnames(t30, "Code", "refcode")
#setnames(t10, "description", "refdesc")
setnames(t20, "description", "refdesc")
setnames(t30, "description", "refdesc")

all01 <- Reduce(function(...) merge(..., all.x = TRUE, by = c("refcode", "refdesc")),
                 list(t40_trn, t10, t20, t30, t50_trn))
```

```
all01 <- all01 [ refcode != "sandhigata vaa"]

all01_trn <- melt (data = all01,
                   id.vars = c("refcode", "refdesc"),
                   measure.vars = c("perc10", "perc20", "perc30",
                                    "perc40_After", "perc40_Before",
                                    "perc50_After", "perc50_Before") )

all01_trn <- as.data.table ( sqldf("select *,
                   case
                                   When variable == 'perc10' then '1 Unique patients'
                                   when variable == 'perc20' then '2 no of times disease
reported'
                                   when variable == 'perc30' then '3 disease chronology'
                                   when variable == 'perc40_After' then '4 no of diseases
before'
                                   when variable == 'perc40_Before' then '5 no of
diseases after'
                                   when variable == 'perc50_After' then '6 no of
medicines before'
```

```
                                    when variable == 'perc50_Before' then '7 no of
medicines after'
                                    end as category
                                    from all01_trn"))


# Possible background creation within Tableau
all01_trn <- all01_trn [, valuedumm := 100,]


# Used for the tableau visual
fwrite(all01_trn, file="D:/Hospital_data/ProgresSQL/analysis/300_radar_plot.csv")
################################################################################
# End of program
################################################################################
```

6.5.45 Analysis program for Figure 3-42

R program: 06_d3tree_diagram.R

```
#####################
# This version is for m / f
# Use 04dis_gender_csv folder
#####################
all_met_rmsd <- readRDS("D:/Hospital_data/ProgresSQL/analysis/01adsl_met_rmsd.rds")
```

```
all_met_rmsd <- all_met_rmsd [, Code := str_replace_all(Code, "\\.", "_")]
cnt<- unique( all_met_rmsd [patient_gender != "" & Code != "",
                            c("mr_no", "studyday","Code", "description","distype",
"patient_gender"), ])
cnt <- cnt [, `:=` (mnth = round( studyday /30.25, digits = 0),
                    description = paste("[", trimws(distype), ": ", trimws(description),
"]", sep = ""),
                    Code =  paste("[", trimws(Code), "]", sep="")) ]
cntrow <- cnt [, .(permnth = uniqueN(Code) ), by =.(mr_no, mnth)]

cnt <- cnt [order(mr_no, studyday, Code, description, patient_gender)]
cnt2 <- unique(cnt [, c("mr_no", "Code", "description", "patient_gender", "distype"), ])

# Combinations for each patient
# Do these calculations for first rows
cnt3 <- cnt2[, `:=` (numcomb = seq_len(.N),
                     descomb = description,
                     discomb = Code,
                     grpcomb = paste(trimws(Code), collapse = " ", sep=)),
             by = .(mr_no, patient_gender)]
```

```
cnt30 <- cnt3 [numcomb > 1, `:=` (discomb = sapply(seq_len(.N), function(x)
paste(Code[seq_len(x)], collapse = ">")),
                                  descomb = sapply(seq_len(.N), function(x)
paste(description[seq_len(x)], collapse = ">")) ),
              by = .(mr_no, patient_gender)]


cnt31 <- rbind(cnt3 [numcomb ==1], cnt30 [numcomb > 1])


# Starting disease sttdis

stt <- cnt3 [ numcomb == 1, .(sttdis = paste(descomb, ">", patient_gender, sep="")), by
=.(mr_no, patient_gender, Code, description)]

cnt3disprgs <- merge(cnt31, stt [, c("mr_no", "sttdis"), ], by = c("mr_no"))


cnt3disprgs <- cnt3disprgs [, .(npt = uniqueN(mr_no)), by = .(discomb, descomb, numcomb,
grpcomb, sttdis, patient_gender)]


cnt3disprgs <- cnt3disprgs [order(sttdis, patient_gender, numcomb, discomb, grpcomb)]

cnt3disprgs <- cnt3disprgs [, node := 1:.N, by =.(sttdis, patient_gender, grpcomb, npt)]

cnt3disprgs <- cnt3disprgs [, treecomb := paste(sttdis, ">", descomb, " (N=", npt, ")",
sep="")]

cnt3disprgs <- cnt3disprgs [order(sttdis, grpcomb, node)]

cnt3disprgs02 <- cnt3disprgs [numcomb > 1]
```

```
# These 2 subsets are for the CSV for D3js
sttdis <- unique(stt [, c("description"), ])
sttdisgen <- unique(stt [ sttdis != "", c("sttdis"), ])
frow <- data.table ( treecomb = "id,value")

# Rename to the same variable
setnames(sttdis, "description", "treecomb")
setnames(sttdisgen, "sttdis", "treecomb")

cnt3disprgs03 <- rbind(cnt3disprgs02 [, c("treecomb")], sttdis, sttdisgen)
[order(treecomb)]
cnt3disprgs03 <- cnt3disprgs03[, treecomb := paste("Disease>", treecomb, sep="")]

# No subset
fwrite(unique(cnt3disprgs03),
       col.names = FALSE,
       quote = FALSE,
       "D:\\Hospital_data\\ProgresSQL\\misc\\jsfolder\\999temp\\decode_gender.csv")
##############################################################################
```

```
# End of program
################################################################################


# Create Json file using the following commands:
# This is the working directory path.
SimpleJar=Hospital_data/ProgresSQL/misc/jsfolder/999temp
java -classpath `cygpath -wp /cygdrive/d/${SimpleJar}:./json-simple-1.1.1.jar` D3Taxonomy
decode_gender.csv ">"
################################################################################
# End of program
################################################################################
```