DASC509: Data and Engineering for Health Research

Assignment 1

Date	30/10/2024
Word Count	794
Student ID number	201857199

Code repository

https://github.com/[redacted]/ULiv_MSC_MedStats/blob/1e4c93af047f6ff57d4200d209b3a62df4078966/DASC509-1/DASC509_Assignment_1.Sql

Generative Artificial Intelligence (GAI)

Please include one of the following declarations for this assignment:

- I **did not** use GAI in the preparation of this work

DASC509 Assessment 1

734 words

> **Question 1: 180 words**
>
> Write a short summary about CPRD data, including the sources of the data and the content generally available. (200 words)

The Clinical Practice Research Datalink (CPRD) is a multi-agency database service provided by the UK Government which aims to make anonymised real-world primary care data gathered by a network of GP practices across the UK accessible to healthcare researchers.

Such data is collected through fully-coded patient electronic health records from practices who have deployed either the ’Vision’ or ’EMIS’ software systems. Typically this dataset includes information on demographic characteristics, diagnoses & symptoms, drug exposures, vaccination history, laboratory tests, referrals to hospital and specialist care with optional links to hospital care, death registry, cancer registry, mental health services among others (1).

Running for about 35 years, this service provides access to a longitudinal, prospective,retrospective and representative sampleset permitting the investigation of effectiveness of health policy, health care delivery, risk factors, drug safety and other associated research themes. Such datasets have been used in high-impact studies such as elucidating higher risks of viral respiratory infections in cancer survivors (2), assessing the safety of the national meningococcal group B vaccine programme (3) or estimating the number of shoulder dislocations in the UK (4).

> **Question 2: 89 words**
>
> Create a list of all the tables in the database giving some basic information for each table:
>
> 1. Row counts.
>
> 2. Number of patients/individuals (where applicable).

Table 1: Count information for all provided tables in database

Table	Unique entries (#)	Total entries (#)
cprdsyn_patient	200	200
cprdsyn_medication	200	7079
cprdsyn_observation	200	973
cprdsyn_gender	4	4
cprdsyn_md	81	81
cprdsyn_patienttype	32	32
cprdsyn_pd	128	205
cprdsyn_practice	14	14
cprdsyn_region	13	13

To obtain this solution, we will initially run a query to determine table schema, specifically column names, data types and if the data in the table might be a null value. This information will directly be pulled from the information schema of the database.

```sql
--https://www.mssqltips.com/sqlservertutorial/183/information-schema-columns/
--https://www.w3schools.com/sql/sql_ref_like.asp
SELECT
table_name,
column_name,
column_default,
is_nullable,
data_type,
character_maximum_length
FROM
INFORMATION_SCHEMA.COLUMNS
WHERE
table_name LIKE 'cprdsyn_%'
ORDER BY
column_name;
```

We next identify and clean missing values where appropriate.

```sql
SELECT * from cprdsyn_pd WHERE drugsubstancename IS NULL;
-- No other tables appear to have rectifiable missing values

UPDATE cprdsyn_pd
SET drugsubstancename = 'Alginic acid'
WHERE prodcodeid = 624841000033110;

UPDATE cprdsyn_pd
SET drugsubstancename = 'Paracetamol'
WHERE prodcodeid = 2711441000033111;
```

We next identify categorisable variables, and count unique categories. For instance, in `cprdsyn_patient`, each patient is unique. However in `cprdsyn_pd` while a drug may have different doses or formulations, the `drugsubstancename` can be used for categorisation.

```sql
SELECT
'cprdsyn_patient' AS Name_Of_Table,
COUNT(DISTINCT(patsid)) AS No_Of_Unique_Rows,
COUNT(patsid) AS No_Of_Total_Rows
FROM cprdsyn_patient
UNION ALL
SELECT
'cprdsyn_medication' AS Name_Of_Table,
COUNT(DISTINCT(patsid)) AS No_Of_Unique_Rows,
COUNT(patsid) AS No_Of_Total_Rows
FROM cprdsyn_medication
UNION ALL
SELECT
'cprdsyn_observation' AS Name_Of_Table,
COUNT(DISTINCT(patsid)) AS No_Of_Unique_Rows,
COUNT(patsid) AS No_Of_Total_Rows
FROM cprdsyn_observation
UNION ALL
SELECT
'cprdsyn_gender' AS Name_Of_Table,
COUNT(DISTINCT(genderid)) AS No_Of_Unique_Rows,
COUNT(genderid) AS No_Of_Total_Rows
FROM cprdsyn_gender
UNION ALL
SELECT
'cprdsyn_md' AS Name_Of_Table,
COUNT(DISTINCT(medcodeid)) AS No_Of_Unique_Rows,
COUNT(medcodeid) AS No_Of_Total_Rows
FROM cprdsyn_md
UNION ALL
SELECT
'cprdsyn_patienttype' AS Name_Of_Table,
COUNT(DISTINCT(patienttypeid)) AS No_Of_Unique_Rows,
COUNT(patienttypeid) AS No_Of_Total_Rows
FROM cprdsyn_patienttype
UNION ALL
SELECT
'cprdsyn_pd' AS Name_Of_Table,
COUNT(DISTINCT(drugsubstancename)) AS No_Of_Unique_Rows,
--Different doses of same drug substance are regarded
--as identical for categorisation
COUNT(prodcodeid) AS No_Of_Total_Rows
FROM cprdsyn_pd
UNION ALL
SELECT
'cprdsyn_practice' AS Name_Of_Table,
COUNT(DISTINCT(pracid)) AS No_Of_Unique_Rows,
COUNT(pracid) AS pracid
FROM cprdsyn_practice
UNION ALL
SELECT
'cprdsyn_region' AS Name_Of_Table,
COUNT(DISTINCT(regionid)) AS No_Of_Unique_Rows,
COUNT(regionid) AS No_Of_Total_Rows
FROM cprdsyn_region
```

> **Question 3: 294 words**
>
> Create summary tables describing the features of the patients in the dataset, including:
>
> 1. Summary statistics for all numerical fields.
> 2. Counts for all categorical or binary fields.
> 3. Earliest and latest dates for all date fields.

We will first specify the following ***exclusion*** criteria for each patient:

1. The data do not have an 'Acceptable' research quality as specified by the CPRD.
2. The patient's date of birth is after any observations/diagnoses
3. The patient's date of registration with the GP is after the date of diagnosis. While this may exclude patients who have switched GP practices, it ensures we capture the entire diagnosis-treatment cycle as much as practicable and we are not relying on partial data from non-CPRD registered GP practices.
4. The patient's earliest prescription is issued before their earliest diagnosis.
5. The patient is prescribed any medicines or diagnosed with any conditions after their death.

While it may occur that an individual accessing the practice as a private patient may receive better care or may be seen more quickly or often, we assume in this analysis that a patient in a GP practice will receive the same care irrespective of if they are temporary, NHS or private patients.

Table 2: Summary statistics for numerical columns associated with patient information from intermediary table DASC5091

Table	Column	Minimum	Maximum	Mean	StDev	Median	IQR	Total Rows	Unique Rows
dasc5091	patient_registered_yrs	1926	2017	1994.907	19.35056	2001	22	107	52
dasc5091	patient_age_yrs	7	102	61.08411	26.14442	68	40	107	63
dasc5091	obs_diagnoses_total	1	3690	221.4953	477.7689	39	196	107	70
dasc5091	medication_duration_total_days	0	47070	4589.673	8927.211	505	4370	107	86
dasc5091	medication_duration_average_days	0	94	24.91589	12.63801	28	9	107	38
dasc5091	medication_prescriptionstotal	1	3690	221.4953	477.7689	39	196	107	70
dasc5091	medication_patientyr	0	615	48.26168	94.11768	14	55	107	55
dasc5091	medication_unique	1	20	4.317757	3.599261	3	3.5	107	14

Table 3: Summary statistics for categorical columns associated with patient information from intermediary table DASC5091

Table	Column	Total Rows	Unique Rows
dasc5091	patient_id	107	107
dasc5091	patient_gender	107	2
dasc5091	patient_region	107	6

Table 4: Summary statistics for dates associated with patient information from intermediary table DASC5091

Table	Column	Earliest	Latest	Earliest YR	Latest YR	Total Rows	Unique Rows
dasc5091	patient_datedob	01/01/1921	01/01/2017	1921	2017	107	76
dasc5091	patient_datedeath	28/03/2014	24/03/2020	2014	2020	10	10
dasc5091	obs_earliestdiagnosis	02/10/1991	07/02/2020	1991	2020	107	104
dasc5091	obs_latestdiagnosis	07/07/1998	28/02/2020	1998	2020	107	104
dasc5091	medication_earliestprescription	16/11/2012	07/02/2020	2012	2020	107	103
dasc5091	medication_latestprescription	06/01/2014	27/04/2020	2014	2020	107	101

To achieve these, we will create two intermediary tables, `DASC5090` which contains raw data and `DASC5091` which will contain summarised data.

To create `DASC5090`, let us remove normalisation to increase table readability. Let us also convert dates to years where we do not need granular detail and discard entries which do not make logical sense (eg. dates of diagnoses before the birth of the individual).

```sql
SELECT
--  Patient data

cprdsyn_patient.patsid::VARCHAR AS Patient_ID,
--  Cast to text as Long numbers may result in loss of precision per CPRD guidance.

cprdsyn_gender.genderid::SMALLINT AS Patient_GenderID,
cprdsyn_gender.description::VARCHAR AS Patient_Gender,

DATE_PART('year',cprdsyn_patient.regstartdate)::INT AS Patient_Registered_YRs,
-- Do not need granular date of reg

cprdsyn_patient.dob AS Patient_DateDOB,
cprdsyn_patient.emis_ddate AS Patient_DateDeath,
CASE WHEN cprdsyn_patient.emis_ddate IS NULL THEN
(current_date - cprdsyn_patient.dob)/365::SMALLINT
ELSE (cprdsyn_patient.emis_ddate - cprdsyn_patient.dob)/365::SMALLINT
```

```sql
END AS Patient_Age_YRs,
--  Calculate age by subtracting current date/Date of death from DOB

cprdsyn_region.regionid::SMALLINT AS Patient_RegionID,
cprdsyn_region.description::VARCHAR AS Patient_Region,

--  Observation data

cprdsyn_observation.obsdate AS Obs_DateDiagnosis,
cprdsyn_md.medcodeid::BIGINT AS Obs_MedcodeID,
cprdsyn_md.cleansedreadcode::VARCHAR AS Obs_CleanedReadcode,

--  Medication data

cprdsyn_medication.issuedate AS Medication_DateIssue,
DATE_PART('year',cprdsyn_medication.issuedate) AS Medication_DateIssue_YRs,

cprdsyn_medication.prodcodeid::BIGINT AS Medication_ProductCode,
CASE WHEN cprdsyn_pd.drugsubstancename IS NULL THEN
cprdsyn_medication.prodcodeid::VARCHAR
ELSE cprdsyn_pd.drugsubstancename::VARCHAR
END AS Medication_IDorName,
-- If drug's active substance is null, replace with product code

cprdsyn_medication.quantity::INT AS Medication_Quantity,
cprdsyn_medication.duration::INT AS Medication_DurationOfTreatment_Days
-- Convert days into years to maintain uniformity? Too small

INTO DASC5090

FROM
cprdsyn_patient
-- Join with Region
LEFT JOIN cprdsyn_practice ON cprdsyn_patient.pracid = cprdsyn_practice.pracid
LEFT JOIN cprdsyn_region ON cprdsyn_practice.region = cprdsyn_region.regionid
-- Join with gender description
LEFT JOIN cprdsyn_gender ON cprdsyn_patient.gender = cprdsyn_gender.genderid
--  Join with patient type description
-- LEFT JOIN cprdsyn_patienttype ON cprdsyn_patient.patienttypeid = cprdsyn_patienttype.patienttypeid
--  Join with observation data
LEFT JOIN cprdsyn_observation ON cprdsyn_patient.patsid = cprdsyn_observation.patsid
LEFT JOIN cprdsyn_md ON cprdsyn_observation.medcodeid = cprdsyn_md.medcodeid
--  Join with medication data
LEFT JOIN cprdsyn_medication ON cprdsyn_patient.patsid = cprdsyn_medication.patsid
LEFT JOIN cprdsyn_pd ON cprdsyn_medication.prodcodeid = cprdsyn_pd.prodcodeid

WHERE cprdsyn_patient.acceptable = 1
```

```
-- 37153 rows remain of research quality data
AND (cprdsyn_observation.obsdate > cprdsyn_patient.dob)
-- 37508 rows remain; Pt can not be diagnosed before birth
AND (cprdsyn_observation.obsdate > cprdsyn_patient.regstartdate) ;
-- 27074 rows remain; Pt cannot be diagnosed before registration with doctor
```

This data is further summarised and refined by:

```
SELECT *
INTO DASC5091
FROM(
SELECT
-- Patient details
DISTINCT patient_id::VARCHAR AS patient_id,
MIN (patient_gender)::VARCHAR AS patient_gender,
-- Distinct(varchar) does not work, but Min(varchar) does? Same value.
MIN (patient_registered_yrs)::INT AS patient_registered_yrs,
MIN (patient_datedob) AS patient_datedob,
MIN (patient_datedeath) AS patient_datedeath,
MIN (patient_age_yrs)::INT AS patient_age_yrs,
MIN (patient_region)::VARCHAR AS patient_region,

-- Observations
MIN (obs_datediagnosis) AS obs_earliestdiagnosis,
MAX (obs_datediagnosis) AS obs_latestdiagnosis,
COUNT(obs_medcodeid)::INT AS obs_diagnoses_total,

-- Medications
MIN (medication_dateissue) AS medication_earliestprescription,
MAX (medication_dateissue) AS medication_latestprescription,
SUM(Medication_Durationoftreatment_DAYS)::INT AS medication_duration_total_DAYs,
AVG(Medication_Durationoftreatment_DAYS)::INT AS medication_duration_average_DAYs,
↪
COUNT(Medication_Productcode)::INT AS medication_prescriptionstotal,
CASE WHEN ((MAX (medication_dateissue)-MIN (medication_dateissue))/365) > 0
THEN COUNT(Medication_Productcode)/
((MAX (medication_dateissue)-MIN (medication_dateissue))/365)
ELSE 0
-- If not forced to 0, it divides by 0.
END AS medication_patientyr,
COUNT(DISTINCT Medication_IDorName)::INT AS medication_unique
FROM DASC5090
GROUP BY patient_id
)
WHERE obs_earliestdiagnosis <= medication_earliestprescription
AND (patient_datedeath IS NULL OR patient_datedeath > obs_latestdiagnosis)
```

```sql
AND (patient_datedeath IS NULL OR patient_datedeath >
↪  medication_latestprescription);
-- Applying remaining exclusion criteria which are based on date summaries
```

Let us script in three functions that summarise the major data-types. These functions create dynamic queries to preserve flexibility, but potentially incur the risk of SQL-injection as it passes a string as an argument directly to a query.

```sql
CREATE OR REPLACE FUNCTION Numeric_Summaries(table_name TEXT, column_name TEXT)
RETURNS TABLE (
Tble_Name TEXT,
Clumn_Name TEXT,
Minimum NUMERIC,
Maximum NUMERIC,
Mean NUMERIC,
Stdev NUMERIC,
Median NUMERIC,
IQR NUMERIC,
No_Of_Rows BIGINT,
No_Of_Unique_Rows BIGINT
) AS $$
DECLARE
SumStats TEXT;
BEGIN
-- Force cast to numeric to prevent mismatches.
SumStats := format(
'SELECT
%L AS Tble_Name,
%L AS Clumn_Name,
MIN(%I)::NUMERIC AS Minimum,
MAX(%I)::NUMERIC AS Maximum,
AVG(%I)::NUMERIC AS Mean,
STDDEV_SAMP(%I)::NUMERIC AS Stdev,
PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY %I)::NUMERIC AS Median,
PERCENTILE_CONT(0.75) WITHIN GROUP (ORDER BY %I)::NUMERIC -
PERCENTILE_CONT(0.25) WITHIN GROUP (ORDER BY %I)::NUMERIC AS IQR,
COUNT(%I) AS No_Of_Rows,
COUNT(DISTINCT(%I)) AS No_Of_Unique_Rows
FROM %I',
table_name, column_name,column_name,column_name, column_name, column_name,
column_name, column_name, column_name,column_name,column_name, table_name
);

RETURN QUERY EXECUTE SumStats;
END;
```

```
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION Cat_Summaries(table_name TEXT, column_name TEXT)
RETURNS TABLE (
Tble_Name TEXT,
Clumn_Name TEXT,
No_Of_Rows BIGINT,
No_Of_Unique_Rows BIGINT
) AS $$
DECLARE
CatStats TEXT;
BEGIN
CatStats := format(
'SELECT
%L AS Tble_Name,
%L AS Clumn_Name,
COUNT(%I) AS No_Of_Rows,
COUNT(DISTINCT(%I)) AS No_Of_Unique_Rows
FROM %I',
table_name, column_name,column_name,column_name, table_name
);
RETURN QUERY EXECUTE CatStats;
END;
$$ LANGUAGE plpgsql;

CREATE OR REPLACE FUNCTION Date_Summaries(table_name TEXT, column_name TEXT)
RETURNS TABLE (
Tble_Name TEXT,
Clumn_Name TEXT,
Earliest DATE,
Latest DATE,
Earliest_Year NUMERIC,
Latest_Year NUMERIC,
No_Of_Rows BIGINT,
No_Of_Unique_Rows BIGINT
) AS $$
DECLARE
DateStats TEXT;
BEGIN
-- Force cast to numeric to prevent mismatches.
DateStats := format(
'SELECT
%L AS Tble_Name,
%L AS Clumn_Name,
MIN(%I)::DATE AS Earliest,
MAX(%I)::DATE AS Latest,
DATE_PART(''year'',MIN(%I))::numeric AS Earliest_Year,
-- Double quote to escape properly
DATE_PART(''year'',MAX(%I))::numeric AS Latest_Year,
COUNT(%I) AS No_Of_Rows,
COUNT(DISTINCT(%I)) AS No_Of_Unique_Rows
FROM %I',
table_name, column_name,column_name,column_name, column_name, column_name,
column_name, column_name,table_name
);
RETURN QUERY EXECUTE DateStats;
END;
$$ LANGUAGE plpgsql;
```

Per the structure of DASC5091 as defined above, Let us then call these functions on the appropriate columns.

```
-- Summaries of numeric columns

SELECT * FROM Numeric_Summaries('dasc5091','patient_registered_yrs')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','patient_age_yrs')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','obs_diagnoses_total')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','medication_duration_total_days')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','medication_duration_average_days')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','medication_prescriptionstotal')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','medication_patientyr')
UNION ALL
SELECT * FROM Numeric_Summaries('dasc5091','medication_unique');

-- Summaries of categorical columns
SELECT * FROM Cat_Summaries('dasc5091', 'patient_id')
UNION ALL
SELECT * FROM Cat_Summaries('dasc5091', 'patient_gender')
UNION ALL
SELECT * FROM Cat_Summaries('dasc5091', 'patient_region');

-- Summaries of Dates
SELECT * FROM Date_Summaries('dasc5091', 'patient_datedob')
UNION ALL
SELECT * FROM Date_Summaries ('dasc5091', 'patient_datedeath')
UNION ALL
SELECT * FROM Date_Summaries('dasc5091', 'obs_earliestdiagnosis')
UNION ALL
SELECT * FROM Date_Summaries('dasc5091', 'obs_latestdiagnosis')
```

```
                UNION ALL
                SELECT * FROM Date_Summaries('dasc5091', 'medication_earliestprescription')
                UNION ALL
                SELECT * FROM Date_Summaries('dasc5091', 'medication_latestprescription');
```

Question 4: 5 words

Create a list of all the unique diagnoses which exist within the dataset.

CPRD	Disease Term	Cleaned	SNOMED	SNOMED
Medcode		Readcode	Concept	Description
146927011	Acne vulgaris	M261000	88616000	146927011
178809013	Acquired hypothyroidism	C04..00	111566002	178809013
18268014	Acute bronchitis	H060.00	10509002	18268014
89308010	Acute conjunctivitis	F4C0.00	53726008	89308010
419211018	Acute exacerbation of asthma	H333.00	708038006	3032747019
94884017	Acute myocardial infarction	G30..00	57054005	94884017
486416017	Acute pharyngitis	H02..00	363746003	486416017
300997012	Acute respiratory infections	H0...00	195647007	300997012
26785019	Acute sinusitis	H01..00	15805002	26785019
29982014	Acute tonsillitis	H03..00	17741008	29982014
399230013	Anaemia unspecified	D21z.00	271737000	406638014
299757012	Angina pectoris	G33..00	194828000	299757012
488211000006112	Anxiety with depression	E200300	231504006	346979010
301485011	Asthma	H33..00	195967001	301485011
497341000006116	Atopic dermatitis/eczema	M111.00	24079001	40423010
82343012	Atrial fibrillation	G573000	49436004	82343012
308368017	Cellulitis NOS	M03z000	128045006	474280013
546411000006111	Chest infection	H06z011	195742007	301131016
396090018	Chest infection NOS	H06z000	50417007	83992015

Continued on next page

(Continued)

546761000006119	Chickenpox - varicella	A52..00	38907003	491830017
304071000000115	Chronic kidney disease stage 3	1Z12.00	433144002	2773184015
475431013	Chronic obstructive pulmonary disease	H3...00	13645005	475431013
17160012	Conjunctivitis	F4C0.12	9826008	17160012
25076018	Constipation	19C..00	14760008	25076018
65119018	Cystitis	K15..00	38822007	65119018
295535012	Depressive disorder NEC	E2B..00	35489007	59212011
121589010	Diabetes mellitus	C10..00	73211009	121589010
264681018	Diabetic on oral treatment	66A4.00	170746002	264681018
103578017	Diarrhoea	19F2.00	62315008	103578017
252560012	Dyspepsia	J16y400	162031009	252560012
399917015	Eczema NOS	M12z100	43116000	71923017
1786154013	Erectile dysfunction	E227311	397803000	2955652011
99042012	Essential hypertension	G20..00	59621000	99042012
762181000006116	Flu like illness	H27z.11	95891005	1235951017
42550011	Gastroenteritis	J43..11	25374005	42550011
40268016	Glaucoma	F45..00	23986001	40268016
150085018	Gout	C34..00	90560007	150085018
501500014	Haemorrhoids	G84..00	70153002	501500014
817341000006110	Hay fever - pollens	H170.11	21719001	481104016
139434018	Hiatus hernia	J34..11	84089009	139434018
293299018	Hyperlipidaemia NOS	C324.00	55822004	497411018

Continued on next page

(Continued)

64168014	Hypertensive disease	G2...00	38341003	64168014
68268011	Hypothyroidism	C04..13	40930008	68268011
80425016	Impetigo	M05..00	48277006	80425016
144257010	Infective otitis externa	F501.00	86981007	144257010
745851000006117	Iron deficiency anaemias	D00..00	87522002	507616014
18666015	Irritable bowel syndrome	J521.11	10743008	18666015
2534664018	Ischaemic heart disease	G3...00	414545008	2534664018
221521000000114	Knee osteoarthritis NOS	N05z611	239873007	359420013
357890015	Leg ulcer NOS	M271.13	95344007	512080011
722361000006115	Malignant neoplasm of prostate	B46..00	399068003	1773293010
1480833015	Menorrhagia	K592000	386692008	1480833015
63055014	Migraine	F26..00	37796009	63055014
693461000006115	Musculoskeletal and connective tissue diseases	N....00	312225001	455899017
2535065012	Obesity	C380.00	414916001	2535065012
1776248011	Osteoarthritis	N05..11	396275006	1776248011
107806013	Osteoporosis	N330.00	64859006	107806013
399496018	Otitis externa NOS	F502z00	3135009	6305018
399498017	Otitis media NOS	F52z.00	65363002	108597015
311385019	Plantar fasciitis	N217900	202882003	311385019
108529013	Polymyalgia rheumatica	N20..00	65323003	108529013
308753015	Psoriasis NOS	M161z00	9014002	15886015
398852019	Pure hypercholesterolaemia	C320.00	267432004	398852019

Continued on next page

(Continued)

116082011	Rheumatoid arthritis	N040.00	69896004	116082011
38727013	Sciatica	N143.00	23056005	38727013
61668014	Sinusitis	H01..11	36971009	61668014
459357017	Suspected UTI	1J4..00	314940005	3038119015
149482010	Tonsillitis	H03..12	90176007	149482010
197761014	Type 2 diabetes mellitus	C10F.00	44054006	197761014
396089010	Upper respiratory infection NOS	H05z.00	54150009	89996011
73091000006118	Upper respiratory tract infection NOS	H05z.11	54150009	89996011
74781000006117	Urinary tract infection, site not specified	K190.00	68566005	113884018
105450017	Verruca plantaris	A781100	63440008	105450017
56807016	Viral illness	A79z.11	34014006	56807016
61301000006115	Viral infection NOS	A79z.00	34014006	56799018
350040017	Viral upper respiratory tract infection NOS	H05z.12	281794004	419887019

This can be executed by:

```sql
SELECT
DISTINCT obs_medcodeid::varchar,
cprdsyn_md.term::varchar,
cprdsyn_md.cleansedreadcode::varchar,
cprdsyn_md.snomedctconceptid::varchar,
cprdsyn_md.snomedctdescriptionid::varchar
FROM dasc5090
LEFT JOIN cprdsyn_md ON dasc5090.obs_medcodeid = cprdsyn_md.medcodeid
WHERE
dasc5090.patient_id IN (SELECT patient_id FROM dasc5091)
-- Apply the exclusion criteria defined in DASC5091
AND obs_medcodeid != 999999999;
```

```
                -- No diagnosis is not a unique diagnosis
```

> **Question 5: 164 words**
>
> Create a table of summary statistics based on the number of medications/treatments prescribed per patient prescribed in each year. The resulting table should have:
>
> 1. Total number of medications prescribed each year
> 2. Total number of patients prescribed at least one medication each year.
> 3. The range (minimum, maximum) of numbers of medications/treatments per patient for each year.
> 4. The mean number of medications/treatments per patient per year.
> 5. The median number of medications/treatments per patient per year.

Table 5: Summary statistics of prescriptions per patient per calendar year

Year	Patients (#)	Min prescriptions (#)	Max prescriptions (#)	Total prescriptions (#)	Mean prescriptions per patient	Median prescriptions per patient
2012	1	4	4	4	4	4
2013	36	3	300	1327	37	19
2014	45	2	396	2797	62	48
2015	50	2	576	3223	64	44.5
2016	53	1	588	3650	69	30
2017	58	1	1146	3951	68	33.5
2018	58	1	792	3794	65	24
2019	55	1	996	3794	69	24
2020	35	1	342	1160	33	16

There are two approaches to solving prescriptions per patient per year: By calculating the prescription per patient per year for each patient individually as done in `DASC5091`'s `medication_patientyr` and calculated by calling `SELECT * FROM Numeric_Summaries('dasc5091','medication_patientyr')` we maintain the fidelity of per patient resolution and each time interval is calculated depending on the patient's first prescription. For instance, a patient taking two prescriptions on `2021-12-31` and `2022-01-01` will be counted as 2 prescriptions/year with a year defined as an interval upto 365 days since the first prescription.

However, this approach precludes the calculation of total prescriptions per year. The propensity of SQL for database management enforces some limitations on such data wrangling. By using the second approach as detailed below, we trade per patient resolution for a per calendar year resolution, enabling us to execute the question to completion. In this approach, the same patient taking two prescriptions on 2021-12-31 and 2022-01-01 will be counted as having 1 prescription/year in both 2021 and 2022.

```sql
SELECT
MIN(Yr),
COUNT(No_patients) AS No_patients,
MIN(medication_prescriptionstotal)::INT AS medication_prescriptionmin,
MAX(medication_prescriptionstotal)::INT AS medication_prescriptionmax,
SUM(medication_prescriptionstotal)::INT AS medication_prescriptiontotal,
AVG(medication_prescriptionstotal)::INT AS medication_ppyavg,
PERCENTILE_CONT(0.5) WITHIN GROUP (ORDER BY medication_prescriptionstotal) AS medication_ppymed
FROM(
SELECT
MIN(medication_dateissue_Yrs)::INT AS yr,
COUNT(DISTINCT patient_id)::INT AS No_patients,
COUNT(Medication_Productcode)::INT AS medication_prescriptionstotal
FROM DASC5090
WHERE DASC5090.patient_id IN (SELECT patient_id FROM DASC5091)
-- Apply the exclusion criteria defined in DASC5091
GROUP BY medication_dateissue_Yrs, patient_id)
GROUP BY yr
ORDER BY yr;

```

References

[1] Herrett, E., Gallagher, A. M., Bhaskaran, K., Forbes, H., Mathur, R., Van Staa, T., and Smeeth, L. *International journal of epidemiology* **44**(3), 827–836 (2015).

[2] Carreira, H., Strongman, H., Peppa, M., McDonald, H. I., dos Santos-Silva, I., Stanway, S., Smeeth, L., and Bhaskaran, K. *EClinicalMedicine* **29** (2020).

[3] Bryan, P., Seabroke, S., Wong, J., Donegan, K., Webb, E., Goldsmith, C., Vipond, C., and Feavers, I. *The Lancet Child & Adolescent Health* **2**(6), 395–403 (2018).

[4] Shah, A., Judge, A., Delmestri, A., Edwards, K., Arden, N. K., Prieto-Alhambra, D., Holt,

T. A., Pinedo-Villanueva, R. A., Hopewell, S., Lamb, S. E., et al. *BMJ open* **7**(11), e016112 (2017).