-- icnarc\_\_05\_\_create\_visit\_occurrence\_table

-- This script creates:

-- 'icnarc\_05a\_with\_row\_id'

-- 'icnarc\_05i\_care\_site\_table' (once lookup table provided and uploaded as 'icnarc\_care\_site\_id\_upload')

-- 'icnarc\_05l\_concept\_id\_table'

-- 'icnarc\_05n\_visit\_occurrence\_table'

-- Note that tables are created with the naming convention

-- icnarc\_XXy\_description, where

-- XX is the number, e.g. '01' relating to the script number

-- y is a letter, e.g. 'a' relating to the order of table creation within the script

-- Tables are dropped when no longer needed

-- (See post-script for an earlier version)

-- Steps:

-- 1) Add a row id number to icnarc table

-- 2) Create a table that logically orders visits, to disambiguate the order when 2+ visits happen on the same day

-- 3) Restructure relevant visit dates into long format and join with the ordered table above

-- 4) Create a table that creates the locations for each visit, then switch commas to semicolons for next step

-- 5) Restructure locations into long format, then join with table created in step 3

-- 6) Manually upload care site lookup table then join them to the visits table

-- 7) Create the care site ID table for the ICNARC dataset

-- 8) Create concept ID lookup table then join with visits table

-- 9) Create the concept ID table for the ICNARC dataset

-- 10) Add to/from to visit\_locations and switch ";" back to ","

-- 11) Convert visit\_start\_datetime from just dates to datetimes where possible (unit\_admission\_datetime and estimated\_unit\_discharge\_datetime)

-- Create visit occurrence table

-- for each relevant date for each person, add a row to the visit occurence table

-- columns should be like:

-- visit\_occurrence\_id (primary key, unique; row number)

-- person\_id

-- icu\_admission\_id (1, 2... unique icu admissions within a person but not uniquely numbered across people)

-- row\_id\_in\_icnarc\_table (a foreign key)

-- visit\_id (1, 2... unique visit id within a person but not uniquely numbered across people)

-- visit\_start\_datetime (the date / datetime of the admission/discharge. Decided with John just to use start dates and not end dates as I only have one end date, plus can make a path through the system if I put everything as start dates)

-- visit\_description (the name of the original column in the icnarc table that housed the date)

-- 1) Add a row id number to icnarc table

-- add row number, ordering by unit\_admission\_date\_time then person\_id

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05a\_with\_row\_id`

AS

SELECT

ROW\_NUMBER() OVER (ORDER BY unit\_admission\_datetime, person\_id) AS icnarc\_row\_id,

\*

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_04e\_unit\_discharge\_status\_estimated`

ORDER BY unit\_admission\_datetime, person\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_04e\_unit\_discharge\_status\_estimated`;

-- 2) Create a table that logically orders visits, to disambiguate the order when 2+ visits happen on the same day

-- make a table with the names of the visit\_descriptions (the relevant date variables) and a numbered list of the logical order. This numbered list means that if two visits happen on the same date, they are automatically placed in a logical order.

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05b\_logical\_visit\_order`

AS

SELECT 01 logical\_visit\_order, 'original\_hosp\_admission\_date' visit\_description UNION ALL

SELECT 02, 'original\_unit\_admission\_date' UNION ALL

SELECT 03, 'hospital\_admission\_date' UNION ALL

SELECT 04, 'unit\_admission\_date' UNION ALL

SELECT 05, 'estimated\_unit\_discharge\_date' UNION ALL

SELECT 06, 'date\_body\_removed' UNION ALL

SELECT 07, 'date\_ult\_unit\_discharge';

-- Decided with Kuldeep and John that the following dates are not visits.

-- SELECT 03, 'actual\_date\_of\_delivery' UNION ALL -- removed from visit table

-- SELECT 04, 'date\_of\_last\_critical\_care' UNION ALL -- removed from visit table

-- SELECT 06, 'date\_1st\_managed\_by\_unit\_team' UNION ALL -- removed from visit table

-- SELECT 07, 'expected\_date\_of\_delivery' UNION ALL -- removed from visit table

-- SELECT 08, 'outreach\_last\_visit\_date' UNION ALL -- removed from visit table

-- SELECT 11, 'date\_of\_first\_critical\_care' UNION ALL -- removed from visit table

-- 3) Restructure relevant visit dates into long format and join with the ordered table above

--Join that table above with the long format table below, order by person\_id, visit\_start\_datetime, then the numbered list so that I can then number them to make the visit ID

-- wide to long

-- from https://stackoverflow.com/questions/47650782/reshape-from-wide-to-long-in-big-query-standard-sql

-- (note that one comment says "Perhaps UNPIVOT() is the right approach these days." - I didn't see this til after I'd built this solution)

-- their date is my person\_id

-- their metric columns (A, B, C, D) are my date columns - visit\_description

-- their value is my date

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05c\_ordered\_visits`

AS

WITH `project.dataset.yourtable` AS (

-- select these columns from the source table

SELECT

person\_id,

icnarc\_row\_id,

-- create a column the consecutively numbers icu admissions for each person

DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY icnarc\_row\_id ASC) AS icu\_visit\_id,

-- Decided with Kuldeep and John that the commented-out dates are not visits.

--actual\_date\_of\_delivery,

--expected\_date\_of\_delivery,

--date\_1st\_managed\_by\_unit\_team,

--date\_of\_last\_critical\_care,

original\_hosp\_admission\_date,

original\_unit\_admission\_date,

--outreach\_last\_visit\_date,

unit\_admission\_date,

hospital\_admission\_date,

date\_body\_removed,

--date\_of\_first\_critical\_care,

date\_ult\_unit\_discharge,

estimated\_unit\_discharge\_date

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05a\_with\_row\_id`

)

-- from the select statement above and the from statement below, return these columns

SELECT

person\_id,

icu\_visit\_id,

-- create a column that consecutively numbers visits within each person and icu visit

ROW\_NUMBER() OVER (PARTITION BY person\_id, icu\_visit\_id

ORDER BY visit\_start\_datetime, logical\_visit\_order\_table.logical\_visit\_order ASC) AS visit\_id,

icnarc\_row\_id,

DATE(visit\_start\_datetime) AS visit\_start\_datetime,

long\_format\_visits.visit\_description,

-- can uncomment the line below to double-check the ordering worked, but don't really want it returned

--logical\_visit\_order\_table.logical\_visit\_order AS logical\_visit\_order

FROM (

SELECT

person\_id,

icnarc\_row\_id,

icu\_visit\_id,

-- convert the date columns to long format

REGEXP\_REPLACE(SPLIT(pair, ':')[OFFSET(0)], r'^"|"$', '') visit\_description,

REGEXP\_REPLACE(SPLIT(pair, ':')[OFFSET(1)], r'^"|"$', '') visit\_start\_datetime

FROM `project.dataset.yourtable` t,

UNNEST(SPLIT(REGEXP\_REPLACE(to\_json\_string(t), r'{|}', ''))) pair

) AS long\_format\_visits

-- join with the ordered visits

JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05b\_logical\_visit\_order` AS logical\_visit\_order\_table

ON logical\_visit\_order\_table.visit\_description = long\_format\_visits.visit\_description

-- remove these strings from visit\_description (these entries are a by-product of converting to long format)

WHERE long\_format\_visits.visit\_description != 'person\_id'

AND long\_format\_visits.visit\_description != 'icnarc\_row\_id'

AND long\_format\_visits.visit\_description != 'icu\_visit\_id'

-- remove these dates - agreed with Kuldeep and John that they are not visits [already removed]

-- AND long\_format\_visits.visit\_description != 'actual\_date\_of\_delivery'

-- AND long\_format\_visits.visit\_description != 'expected\_date\_of\_delivery'

-- AND long\_format\_visits.visit\_description != 'date\_of\_last\_critical\_care'

-- AND long\_format\_visits.visit\_description != 'date\_1st\_managed\_by\_unit\_team'

-- AND long\_format\_visits.visit\_description != 'outreach\_last\_visit\_date'

-- AND long\_format\_visits.visit\_description != 'date\_of\_first\_critical\_care'

-- if a date is null, don't include it here

AND visit\_start\_datetime != 'null'

ORDER BY person\_id, icu\_visit\_id, visit\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05b\_logical\_visit\_order`;

-- 4) Create a table that creates the locations for each visit

-- create a table that grabs person id, row number, then creates a definitive location for each visit by merging fields where needed (icnarc v2 and v3 admission software has different field names for the locations)

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05d\_wide\_locations`

AS

SELECT

person\_id,

icnarc\_row\_id,

-- 1/4) LOC\_FOR\_original\_hosp\_admission\_date

CASE

WHEN

-- if both are null, put null

prior\_transfer\_in IS NULL AND transferred\_from IS NULL

THEN NULL

WHEN

-- if only the first one has a value, just use that

prior\_transfer\_in IS NOT NULL AND transferred\_from IS NULL

THEN prior\_transfer\_in

WHEN

-- if only the second one has a value, just use that

prior\_transfer\_in IS NULL AND transferred\_from IS NOT NULL

THEN transferred\_from

WHEN

-- if both have values, use transferred from, if they're not the same

-- originally concatenated these, but this only happens twice and it's prior = BRI then from = Yorkshire (makes sense to put Yorkshire)

-- then the other prior = Airedale General Hospital HDU and from = Airedale General Hospital neither ICU nor HDU

-- so just using transferred from for easier care\_site matching

prior\_transfer\_in IS NOT NULL AND transferred\_from IS NOT NULL AND prior\_transfer\_in != transferred\_from

THEN transferred\_from

WHEN

-- if both have values, if they are the same, just use 1

prior\_transfer\_in IS NOT NULL AND transferred\_from IS NOT NULL AND prior\_transfer\_in = transferred\_from

THEN prior\_transfer\_in

END AS LOC\_FOR\_original\_hosp\_admission\_date,

-- 2/4) LOC\_FOR\_unit\_admission\_date

CASE

WHEN

-- if source of unit admission is not null (it's a v2 admission) use just that

source\_of\_unit\_admission IS NOT NULL

--THEN CONCAT("FROM- " , source\_of\_unit\_admission)

THEN source\_of\_unit\_admission

WHEN

-- if location and hosp housing location aren't null, concatenate them

location IS NOT NULL AND hospital\_housing\_location IS NOT NULL

--THEN CONCAT("FROM- ", location, ", ", hospital\_housing\_location)

THEN CONCAT(location, ", ", hospital\_housing\_location)

WHEN

-- if only location is not null, use that

source\_of\_unit\_admission IS NULL AND location IS NOT NULL AND hospital\_housing\_location IS NULL

THEN location

WHEN

-- if everything is null, use that

source\_of\_unit\_admission IS NULL AND location IS NULL AND hospital\_housing\_location IS NULL

THEN NULL

END AS LOC\_FOR\_unit\_admission\_date,

-- 3/4) LOC\_FOR\_estimated\_unit\_discharge\_date

CASE

WHEN

-- if destination is not null (it's a v2 admission) use just that

destination IS NOT NULL

--THEN CONCAT("TO- " , destination)

THEN destination

WHEN

-- if location\_1 and hosp housing location\_1 aren't null, concatenate them

location\_1 IS NOT NULL AND hospital\_housing\_location\_1 IS NOT NULL

THEN CONCAT(location\_1, ", ", hospital\_housing\_location\_1)

WHEN

-- if only location\_1 is not null, use that

destination IS NULL AND location\_1 IS NOT NULL AND hospital\_housing\_location\_1 IS NULL

THEN location\_1

WHEN

-- if everything is null, use that

destination IS NULL AND location\_1 IS NULL AND hospital\_housing\_location\_1 IS NULL

THEN NULL

END AS LOC\_FOR\_estimated\_unit\_discharge\_date,

-- 4/4) LOC\_FOR\_date\_ult\_unit\_discharge

CASE

WHEN

-- if all are null, put null

ult\_unit\_identifier\_out IS NULL AND ult\_hospital\_trans\_to IS NULL AND transferred\_to IS NULL

THEN NULL

WHEN

-- if only the first one has a value, just use that

ult\_unit\_identifier\_out IS NOT NULL AND ult\_hospital\_trans\_to IS NULL AND transferred\_to IS NULL

THEN ult\_unit\_identifier\_out

WHEN

-- if only the second one has a value, just use that

ult\_unit\_identifier\_out IS NULL AND ult\_hospital\_trans\_to IS NOT NULL AND transferred\_to IS NULL

THEN ult\_hospital\_trans\_to

WHEN

-- if only the third one has a value, just use that

ult\_unit\_identifier\_out IS NULL AND ult\_hospital\_trans\_to IS NULL AND transferred\_to IS NOT NULL

THEN transferred\_to

WHEN

-- if the first says non-icnarc hospital, use only the second

ult\_unit\_identifier\_out = " Non-ICNARC Hospital"

THEN ult\_hospital\_trans\_to

END AS LOC\_FOR\_date\_ult\_unit\_discharge,

"Bradford Royal Infirmary" AS LOC\_FOR\_hospital\_admission\_date,

NULL AS LOC\_FOR\_original\_unit\_admission\_date,

NULL AS LOC\_FOR\_date\_body\_removed

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05a\_with\_row\_id`;

-- substitute the commas in the locations with semicolons so that I can change from long to wide (commas make it think there's multiple values in one cell)

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05e\_wide\_locations\_semicolons`

AS

SELECT

person\_id,

icnarc\_row\_id,

REPLACE(LOC\_FOR\_original\_hosp\_admission\_date, ",", ";") AS original\_hosp\_admission\_date,

REPLACE(LOC\_FOR\_unit\_admission\_date, ",", ";") AS unit\_admission\_date,

REPLACE(LOC\_FOR\_estimated\_unit\_discharge\_date, ",", ";") AS estimated\_unit\_discharge\_date,

REPLACE(LOC\_FOR\_date\_ult\_unit\_discharge, ",", ";") AS date\_ult\_unit\_discharge,

LOC\_FOR\_hospital\_admission\_date AS hospital\_admission\_date,

-- use the original hospital location as the location for the original unit location because sometimes only original unit admission date (and not original hospital admission date) is listed

REPLACE(LOC\_FOR\_original\_hosp\_admission\_date, ",", ";") AS original\_unit\_admission\_date,

LOC\_FOR\_date\_body\_removed AS date\_body\_removed

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05d\_wide\_locations`;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05d\_wide\_locations`;

-- 5) Restructure locations into long format, then join with table created in step 3

-- make the locations longer like I did with dates

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05f\_long\_locations`

AS

SELECT

person\_id,

icnarc\_row\_id,

location\_type,

NULLIF(locations\_table.visit\_location, "null") AS visit\_location

FROM (

SELECT

person\_id,

icnarc\_row\_id,

-- convert the location columns to long format

REGEXP\_REPLACE(SPLIT(pair, ':')[OFFSET(0)], r'^"|"$', '') location\_type,

REGEXP\_REPLACE(SPLIT(pair, ':')[OFFSET(1)], r'^"|"$', '') visit\_location

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05e\_wide\_locations\_semicolons` t,

UNNEST(SPLIT(REGEXP\_REPLACE(to\_json\_string(t), r'{|}', ''))) pair

) as locations\_table

-- remove these strings from visit\_description (these entries are a by-product of converting to long format)

WHERE location\_type != 'person\_id'

AND location\_type != 'icnarc\_row\_id'

ORDER BY

person\_id,

icnarc\_row\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05e\_wide\_locations\_semicolons`;

-- join on the locations to the icnarc\_05c\_ordered\_visits table made earlier

-- also add visit\_occurrence\_id - this is not the chronological order of all visits. It orders by the icnarc\_row\_id (where each icu admission was ordered by datetime then person id), then the visit\_id. So in a way it's ordered by icu admission datetime, partitioned by person id and visit id.

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05g\_visits\_with\_locations`

AS

SELECT

ROW\_NUMBER() OVER (ORDER BY visits\_table.icnarc\_row\_id, visits\_table.visit\_id)

AS visit\_occurrence\_id,

visits\_table.person\_id,

visits\_table.icu\_visit\_id,

visits\_table.visit\_id,

visits\_table.icnarc\_row\_id,

visits\_table.visit\_start\_datetime,

visits\_table.visit\_description,

NULLIF(locations\_table.visit\_location, "null") AS visit\_location

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05c\_ordered\_visits` AS visits\_table

JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05f\_long\_locations` AS locations\_table

ON locations\_table.person\_id = visits\_table.person\_id

AND locations\_table.icnarc\_row\_id = visits\_table.icnarc\_row\_id

AND locations\_table.location\_type = visits\_table.visit\_description

ORDER BY

visits\_table.icnarc\_row\_id,

visits\_table.visit\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05c\_ordered\_visits`;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05f\_long\_locations`;

-- 6) Manually upload care site lookup table then join them to the visits with locations table

-- John to send an updated lookup table - commented out for now

-- join onto visit table with locations above

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05h\_visits\_with\_care\_site\_ids`

-- AS

-- SELECT

-- visits\_table.\*,

-- site\_table.care\_site\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05g\_visits\_with\_locations` AS visits\_table

-- LEFT JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_care\_site\_id\_upload` AS site\_table

-- ON visits\_table.visit\_location = site\_table.name

-- ORDER BY

-- visits\_table.person\_id,

-- visits\_table.icu\_visit\_id,

-- visits\_table.visit\_id;

-- this is temporary code while I cannot run the commented-out section above

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05h\_visits\_with\_care\_site\_ids`

AS

SELECT

\*,

NULL as care\_site\_id

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05g\_visits\_with\_locations`

ORDER BY

visit\_occurrence\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05g\_visits\_with\_locations`;

-- 7) Create the care site ID table for the ICNARC dataset

-- Waiting to receive care site lookup table from John

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05i\_care\_site\_table`

-- AS

-- SELECT DISTINCT

-- master\_care\_sites.\*

-- FROM `yhcr-prd-phm-bia-core.CY\_FDM\_MASTER.care\_site` AS master\_care\_sites

-- JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05h\_visits\_with\_care\_site\_ids` AS icnarc\_care\_sites

-- ON master\_care\_sites.care\_site\_id = icnarc\_care\_sites.care\_site\_id;

-- 8) Create concept ID lookup table then join with visits table

-- create temp concept id table

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05j\_concept\_id\_lookup`

AS

SELECT 8715 concept\_id, 'original\_hosp\_admission\_date' visit\_description UNION ALL

SELECT 4138933, 'original\_unit\_admission\_date' UNION ALL

SELECT 8715, 'hospital\_admission\_date' UNION ALL

SELECT 4138933, 'unit\_admission\_date' UNION ALL

SELECT 45878782, 'estimated\_unit\_discharge\_date' UNION ALL

SELECT NULL, 'date\_body\_removed' UNION ALL

SELECT 45878782, 'date\_ult\_unit\_discharge';

-- add concept IDs to visits table

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05k\_visits\_with\_concepts`

AS

SELECT

visits\_table.visit\_occurrence\_id,

visits\_table.person\_id,

visits\_table.icu\_visit\_id,

visits\_table.visit\_id,

visits\_table.icnarc\_row\_id,

visits\_table.visit\_start\_datetime,

visits\_table.visit\_description,

temp\_concepts.concept\_id,

visits\_table.visit\_location,

visits\_table.care\_site\_id

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05j\_concept\_id\_lookup` temp\_concepts

JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05h\_visits\_with\_care\_site\_ids` AS visits\_table

ON visits\_table.visit\_description = temp\_concepts.visit\_description

ORDER BY

visit\_occurrence\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05h\_visits\_with\_care\_site\_ids`;

-- 9) Create the concept ID table for the ICNARC dataset

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05l\_concept\_id\_table`

AS

SELECT DISTINCT

master\_concepts.\*

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05j\_concept\_id\_lookup` AS temp\_concepts

JOIN `yhcr-prd-phm-bia-core.CY\_CDM\_VOCAB.concept` AS master\_concepts

ON temp\_concepts.concept\_id = master\_concepts.concept\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05j\_concept\_id\_lookup`;

-- 10) Add to/from to visit\_locations and switch ";" back to ","

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05m\_visits\_semicolons\_replaced`

AS

SELECT

\*

REPLACE(

CASE

-- add "from"

WHEN visit\_description = "unit\_admission\_date" OR visit\_description = "date\_ult\_unit\_discharge"

THEN CONCAT("FROM: ", REPLACE(visit\_location, ";", ","))

-- add "to"

WHEN visit\_description = "estimated\_unit\_discharge\_date" OR visit\_description = "original\_hosp\_admission\_date" OR visit\_description = "hospital\_admission\_date"

THEN CONCAT("TO: ", REPLACE(visit\_location, ";", ","))

ELSE

REPLACE(visit\_location, ";", ",")

END AS visit\_location

)

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05k\_visits\_with\_concepts`

ORDER BY

visit\_occurrence\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05k\_visits\_with\_concepts`;

-- 11) Convert visit\_start\_datetime from just dates to datetimes where possible (unit\_admission\_datetime and estimated\_unit\_discharge\_datetime)

-- update dates to date times...

CREATE TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05n\_visit\_occurrence\_table`

AS

SELECT

visits\_table.visit\_occurrence\_id,

visits\_table.person\_id,

visits\_table.icu\_visit\_id,

visits\_table.visit\_id,

visits\_table.icnarc\_row\_id,

-- replace the dates with date times for unit\_admission\_date and estimated\_unit\_discharge\_datetime

CASE

WHEN visit\_description = "unit\_admission\_date"

THEN source\_table.unit\_admission\_datetime

WHEN visit\_description = "estimated\_unit\_discharge\_date"

THEN source\_table.estimated\_unit\_discharge\_datetime

ELSE DATETIME(CONCAT(visits\_table.visit\_start\_datetime, "T00:00:00"))

END AS visit\_start\_datetime,

-- replace the visit descriptions to reflect this

CASE

WHEN visits\_table.visit\_description = "unit\_admission\_date"

THEN "unit\_admission\_datetime"

WHEN visits\_table.visit\_description = "estimated\_unit\_discharge\_date"

THEN "estimated\_unit\_discharge\_datetime"

ELSE visits\_table.visit\_description END AS visit\_description,

visits\_table.concept\_id,

visits\_table.visit\_location,

visits\_table.care\_site\_id

FROM `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05m\_visits\_semicolons\_replaced` visits\_table

JOIN `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05a\_with\_row\_id` source\_table

ON visits\_table.icnarc\_row\_id = source\_table.icnarc\_row\_id

ORDER BY visits\_table.visit\_occurrence\_id;

DROP TABLE `yhcr-prd-phm-bia-core.CY\_MYSPACE\_EmW.icnarc\_05m\_visits\_semicolons\_replaced`;

--================

-- Post-script

--================

-- This is an earlier version of the visit occurrence table which was much simpler but did not enable a 'path through the system'

-- -- Create a visit occurence table which contains:

-- -- Person ID

-- -- Datetime

-- -- Visit ID

-- -- Visit ID should also be added to the source table

-- -- For some datasets, multiple observations/rows will make up one visit

-- -- For ICNARC, likely that each is a separate visit,

-- -- but operationalise visit as each unique combination of Person ID and Datetime

-- -- Should this be done after it's been FDMed?

-- -- Yes, use the CY\_EMILY\_TEST tables

-- -- (though will be moving these once full data dictionary and FDM process confirmed)

-- -- So within each Person ID, consecutively number their visits, unless there are multiple visits on the same day

-- -- Person ID = 1, Date = 04/06/2001, Visit ID = 1

-- -- Person ID = 1, Date = 10/06/2001, Visit ID = 2

-- -- Person ID = 1, Date = 10/06/2001, Visit ID = 2

-- -- Person ID = 1, Date = 30/07/2003, Visit ID = 3

-- -- Person ID = 2, Date = 05/10/2002, Visit ID = 1

-- -- Person ID = 3, Date = 15/03/2007, Visit ID = 1

-- -- Person ID = 3, Date = 18/02/2008, Visit ID = 2

-- -- Person ID = 3, Date = 18/02/2008, Visit ID = 2

-- -- This would be so easier in R, but in Big Query it's probably a case when, or maybe a group by

-- -- Person ID = person\_id

-- -- Date = unit\_admission\_date

-- -- From https://cloud.google.com/bigquery/docs/reference/standard-sql/numbering\_functions

-- -- WITH finishers AS

-- -- (SELECT 'Sophia Liu' as name,

-- -- TIMESTAMP '2016-10-18 2:51:45' as finish\_time,

-- -- 'F30-34' as division

-- -- UNION ALL SELECT 'Lisa Stelzner', TIMESTAMP '2016-10-18 2:54:11', 'F35-39'

-- -- UNION ALL SELECT 'Nikki Leith', TIMESTAMP '2016-10-18 2:59:01', 'F30-34'

-- -- UNION ALL SELECT 'Lauren Matthews', TIMESTAMP '2016-10-18 3:01:17', 'F35-39'

-- -- UNION ALL SELECT 'Desiree Berry', TIMESTAMP '2016-10-18 3:05:42', 'F35-39'

-- -- UNION ALL SELECT 'Suzy Slane', TIMESTAMP '2016-10-18 3:06:24', 'F35-39'

-- -- UNION ALL SELECT 'Jen Edwards', TIMESTAMP '2016-10-18 3:06:36', 'F30-34'

-- -- UNION ALL SELECT 'Meghan Lederer', TIMESTAMP '2016-10-18 2:59:01', 'F30-34')

-- -- SELECT name,

-- -- finish\_time,

-- -- division,

-- -- RANK() OVER (PARTITION BY division ORDER BY finish\_time ASC) AS finish\_rank

-- -- FROM finishers;

-- -- For mine: (RANK skips if there are ties, whereas DENSE\_RANK doesn't)

-- -- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY unit\_admission\_date ASC) AS visit\_id

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_with\_visit\_id`;

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.visit\_occurrence`;

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id`;

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id2`;

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id3`;

-- DROP TABLE IF EXISTS `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id4`;

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id`

-- AS

-- SELECT

-- person\_id,

-- unit\_admission\_date,

-- unit\_admission\_time,

-- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY unit\_admission\_date ASC) AS visit\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`

-- ORDER BY person\_id, unit\_admission\_date;

-- -- Looks like this has worked great for multiple different-day visits per person, but how did it handle same-day visits?

-- -- Let's check this

-- --From:https://stackoverflow.com/questions/57200837/how-to-display-in-big-query-only-duplicated-records

-- SELECT \*

-- EXCEPT(flag) FROM (

-- SELECT \*, COUNT(1) OVER(PARTITION BY person\_id, unit\_admission\_date) > 1 flag

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id`

-- )

-- WHERE flag

-- ORDER BY person\_id, unit\_admission\_date, unit\_admission\_time;

-- -- Yep, it treated multiple occurences on the same day for a given person as the same visit

-- -- There are 8 people who have multiple observations in one day (all 2 observations)

-- -- Maybe worth checking these admissions as it may not make sense for the same person to be admitted twice in one day

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id2`

-- AS

-- SELECT

-- person\_id,

-- unit\_admission\_date,

-- unit\_admission\_time,

-- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY unit\_admission\_date ASC) AS visit\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`

-- ORDER BY person\_id, unit\_admission\_date;

-- SELECT \*

-- EXCEPT(flag) FROM (

-- SELECT \*, COUNT(1) OVER(PARTITION BY person\_id, unit\_admission\_date) > 1 flag

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id2`

-- )

-- WHERE flag

-- ORDER BY person\_id, unit\_admission\_date, unit\_admission\_time;

-- -- For all-but-one person, these admissions happened at different times, so that makes sense

-- -- For one person (319361), they were admitted twice on 2010-11-24 at 18:00:00

-- -- Check this out

-- SELECT \*

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`

-- WHERE

-- person\_id = 319361

-- AND

-- unit\_admission\_date = '2010-11-24';

-- -- These are different people! One's a 32 year old man, one's a 53 year old woman. Clearly a clerical error that they typed the wrong NHS number the second time.

-- -- Let's check the other same-day admissions

-- -- Don't only select 3 columns - select all

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id3`

-- AS

-- SELECT

-- \*,

-- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY unit\_admission\_date ASC) AS visit\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`

-- ORDER BY person\_id, unit\_admission\_date;

-- -- select where there are duplicates of the combination of person\_id and unit\_admission\_date

-- SELECT \*

-- EXCEPT(flag) FROM (

-- SELECT \*, COUNT(1) OVER(PARTITION BY person\_id, unit\_admission\_date) > 1 flag

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id3`

-- )

-- WHERE flag

-- ORDER BY person\_id, unit\_admission\_date, unit\_admission\_time;

-- -- For 7 of the 8 person IDs, there's a different DOB each time

-- -- Check for each person ID, how many different DOBs have been entered

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id4`

-- AS

-- SELECT

-- person\_id,

-- date\_of\_birth,

-- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY date\_of\_birth ASC) AS dob\_num,

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`

-- ORDER BY person\_id, date\_of\_birth;

-- SELECT

-- person\_id,

-- MAX(dob\_num) AS num\_dobs

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id4`

-- GROUP BY person\_id

-- HAVING num\_dobs > 1

-- ORDER BY person\_id;

-- -- There are 19 person\_ids with more than one dob present in the icnarc post-fdm table

-- -- Tell Kuldeep and John about this, but first continue with the visit occurrence stuff

-- -- Delete the wip tables

-- drop table `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id`;

-- drop table `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id2`;

-- drop table `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id3`;

-- drop table `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_wip\_visit\_id4`;

-- -- Add visit id column to icnarc table

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_with\_visit\_id`

-- AS

-- SELECT

-- \*,

-- DENSE\_RANK() OVER (PARTITION BY person\_id ORDER BY unit\_admission\_date ASC) AS visit\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_renamed\_columns`;

-- -- To create the visit\_occurrence table, select just the 3 columns and keep only unique entries

-- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.visit\_occurrence`

-- AS

-- SELECT DISTINCT

-- person\_id,

-- unit\_admission\_date AS visit\_start\_datetime,

-- visit\_id

-- FROM `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_with\_visit\_id`

-- ORDER BY person\_id, visit\_id;

-- -- Done!

-- -- Add a column to the fdm to say whether the DOB matches the person table

-- -- CREATE TABLE `yhcr-prd-phm-bia-core.CY\_EMILY\_TEST.icnarc\_dob\_matches`

-- -- AS

-- -- SELECT \*,

-- -- CASE WHEN icnarc\_data.date\_of\_birth = person.

-- -- change all the DOBS in ICNARC to 15 for the day

-- -- then match to the person table