**Module 1. Retrieving the Diseases (D) and Phenotypic Series (PS) from OMIM**

/\* Retrieve the D and their grouping into PS from the most recent **morbidmap.txt** and **phenotypicSeries.txt** files othe OMIM website. \*/

.open c:/sqlite/LHPS/lhps.db

.mode tabs

.headers on

/\* Import **morbidmap.txt** and select the columns of interest (i.e., D as phenotype name and gene as MIM number) into table tmp2\_morbidmap. \*/

.import c:/sqlite/LHPS/inputs/**morbidmap.txt** tmp1\_morbidmap

CREATE TABLE **tmp2\_morbidmap**(mim\_dgp\_id INT, mim\_name TEXT);

INSERT INTO tmp2\_morbidmap

SELECT DISTINCT \* FROM(SELECT MIM\_number, Phenotype FROM tmp1\_morbidmap);

/\* From each phenotype (i.e., D as mim\_name), extract the leftmost character and the key, to retain only the real D (i.e., first character in string is different from ‘?’, ‘[‘ or ‘{‘) and the key is 3 because the molecular phenotype is known). \*/

ALTER TABLE tmp2\_morbidmap ADD COLUMN start TEXT;

UPDATE tmp2\_morbidmap SET start = SUBSTR(mim\_name, 1,1);

ALTER TABLE tmp2\_morbidmap ADD COLUMN key TEXT;

UPDATE tmp2\_morbidmap SET key = SUBSTR(mim\_name, -3,3);

/\* Delete the following disease phenotypes: (i) susceptibility to multi-factorial disorders (start = ‘{‘); (ii) non-diseases (start = ‘[‘); (iii) unconfirmed diseases (start = ‘?‘). \*/

DELETE FROM tmp2\_morbidmap WHERE start = ‘{’;

DELETE FROM tmp2\_morbidmap WHERE start = ‘[’;

DELETE FROM tmp2\_morbidmap WHERE start = ‘?’;

/\* Delete the following D: (i) D with unknown gene defect (key = ‘(1)‘); (ii) D with unknown mutation (key = ‘(2)‘); (iii) D caused by multiple genes (key = ‘(4)‘). This way, we retain only D with known molecular basis (key = 3). \*/

DELETE FROM tmp2\_morbidmap WHERE key = ‘(1)’;

DELETE FROM tmp2\_morbidmap WHERE key = ‘(2)’;

DELETE FROM tmp2\_morbidmap WHERE key = ‘(4)’;

/\* As the Disease Gene Products (DGP) are designated in OMIM with MIM identifiers, retrieve the DGP as entrez(NCBI)/symbol(HUGO). Then, to convert the DGP from OMIM id (‘mim\_dgp\_id’) to Entrez id (entrez\_id), first import the mim2gene file from OMIM. \*/

.import c:/sqlite/LHPS/inputs/**mim2gene.txt** tmp\_mim2gene

CREATE TABLE **mim2gene**(dgp\_mim INT, type TEXT, entrez\_id INT, symbol TEXT);

INSERT INTO mim2gene

SELECT DISTINCT \* FROM(SELECT Mim\_Number, Type, Entrez, Symbol FROM tmp\_mim2gene);

/\* From mim2gene, delete the rows in which the D has been either moved or removed. \*/

DELETE FROM mim2gene WHERE type = ‘moved/removed’;

/\*.Join the D (from tmp2\_morbidmap) with the DGP (from mim2gene) to retrieve the D that have an entrez\_id. \*/

CREATE TABLE **tmp3\_morbidmap**(mim\_dgp\_id INT, mim\_name TEXT, mim\_type TEXT, entrez\_id INT);

INSERT INTO tmp3\_morbidmap

SELECT DISTINCT \* FROM(

SELECT l.mim\_dgp\_id, l.mim\_name, r.type, r.entrez\_id

FROM tmp2\_morbidmap l

LEFT JOIN mim2gene r

ON l.mim\_dgp\_id = r.dgp\_mim

);

/\* Retain the DGP that are protein-coding genes. Join the D (from tmp3\_morbidmap) with the symbols (and type of gene) from the (ftp version of) Entrez Gene, limited to *H. sapiens* (taxon 9606). \*/

.import c:/sqlite/LHPS/inputs/**Homo\_sapiens.gene\_info.txt** tmp\_hs

DELETE FROM tmp\_hs WHERE tax\_id != ‘9606’; /\* Retain H. sapiens only. \*/

CREATE TABLE **hs\_entrez**(entrez\_id INT, symbol TEXT, dbXrefs TEXT, gene\_type TEXT);

INSERT INTO hs\_entrez

SELECT DISTINCT \* FROM(

SELECT GeneID, Symbol, SUBSTR(dbXrefs,1,10), type\_of\_gene FROM tmp\_hs);

/\* Retain only the xrefs to OMIM, in format ‘MIM:XXXXXX’. \*/

UPDATE hs\_entrez SET dbXrefs = NULL WHERE SUBSTR(dbXrefs,1,3) <> ‘MIM’;

/\* Convert the xrefs format ‘MIM:XXXXXX’ into ‘XXXXXX’. \*/

UPDATE hs\_entrez SET dbXrefs =

REPLACE(dbXrefs,dbXrefs,SUBSTR(dbXrefs,5,10)) WHERE dbXrefs IS NOT NULL;

/\* As mentioned above, join the D (from tmp3\_morbidmap) with the symbols (from hs\_entrez). \*/

CREATE TABLE **tmp4\_morbidmap**(mim\_name TEXT, dgp\_mim INT, entrez\_id INT, symbol TEXT, gene\_type TEXT, dbXrefs INT);

INSERT INTO tmp4\_morbidmap

SELECT DISTINCT \* FROM(

SELECT l.mim\_name, l.mim\_dgp\_id, l.entrez\_id, r.symbol, r.gene\_type, r.dbXrefs

FROM tmp3\_morbidmap l

LEFT JOIN hs\_entrez r

ON l.entrez\_id = r.entrez\_id

);

/\* Delete the rows in which the gene is NOT a protein-coding gene (e.g., ncRNA, other, snRNA, snoRNA, pseudo, unknown). Also, delete the few D without a corresponding gene (e.g., ‘Myelodysplasia syndrome-1’ 600049, ‘Mulchandani-Bhoj-Conlin syndrome-1’ 617352 and ‘Histiocytosis-lymphadenopathy plus syndrome’ 602782). Note that there might be a discrepancy in the DGP count as dgp\_mim v. entrez\_id/symbol, as one gene (6473 in entrez) corresponds to two items in OMIM, mim 312865 (SHOX) and mim 400020 (SHOY). \*/

DELETE FROM tmp4\_morbidmap WHERE gene\_type != ‘protein-coding’;

DELETE FROM tmp4\_morbidmap WHERE entrez\_id = ‘’;

/\* Retrieve the phenotype (D) id, where available. Check and make correction (e.g., 167400). Then, delete the phenotypes in which the rightmost characters are not 6-digit id. \*/

UPDATE tmp4\_morbidmap SET mim\_name = ‘Paroxysmal extreme pain disorder, 167400 (3)’ WHERE mim\_name = ‘Paroxysmal extreme pain disorder, 167400, (3)’;

/\* Then, remove the rightmost substring “ (3)”. \*/

UPDATE tmp4\_morbidmap SET mim\_name = SUBSTR(mim\_name,1,LENGTH(mim\_name)-4);

/\* Retrieve the rightmost substring of the mim\_name, which contains the phen\_id. Where a phen-id is available, the outcome is a 6-digit integer. Otherwise, the outcome is a non-integer value that will be filtered out in omim-def when setting phen\_id < = 999999, i.e., a non-numeric value). \*/

ALTER TABLE tmp4\_morbidmap ADD COLUMN phen\_id INT;

UPDATE tmp4\_morbidmap SET phen\_id = SUBSTR(mim\_name,-6,6);

UPDATE tmp4\_morbidmap SET phen\_id = NULL WHERE phen\_id > 999999;

/\* Table **omim\_def** is the final version of the morbid map comprehensive of the DGP. \*/

CREATE TABLE **omim\_def**(phen\_mim INT, mim\_name TEXT, dgp\_mim INT, entrez\_id INT, symbol TEXT);

INSERT INTO omim\_def

SELECT DISTINCT \* FROM(

SELECT phen\_id, mim\_name, dgp\_mim, entrez\_id, symbol FROM tmp4\_morbidmap

WHERE phen\_id IS NOT NULL

ORDER BY phen\_id ASC, mim\_name ASC

);

UPDATE omim\_def SET mim\_name = SUBSTR(mim\_name,1,LENGTH(mim\_name)-8);

.once c:/sqlite/LHPS/outputs/**omim\_def.txt**

SELECT \* FROM omim\_def;

DROP TABLE tmp1\_morbidmap; DROP TABLE tmp2\_morbidmap;

DROP TABLE tmp3\_morbidmap; DROP TABLE tmp4\_morbidmap;

DROP TABLE tmp\_mim2gene; DROP TABLE tmp\_hs; DROP TABLE mim2gene;

DROP TABLE hs\_entrez;

/\* Import the PS list as phenotypicSeries.txt (available from OMIM upon request). \*/

.import c:/sqlite/LHPS/inputs/**phenotypicSeries.txt** tmp\_ps

/\* To extract the PS titles from tmp\_ps, retain the rows where column mim\_name - the name of the PS – is empty (‘’), because the PS name is reported in column mim\_id). \*/

CREATE TABLE **ps\_names**(ps\_id TEXT, ps\_name TEXT);

INSERT INTO ps\_names

SELECT DISTINCT \* FROM(SELECT ps\_id, mim\_id FROM tmp\_ps WHERE mim\_name = ‘’);

/\* Remove from tmp\_ps the rows containing just the PS title (as explained above) and not a D within the PS. \*/

DELETE FROM tmp\_ps WHERE mim\_name = ‘’;

CREATE TABLE **ps**(ps\_id TEXT, mim\_id INT, mim\_name TEXT);

INSERT INTO ps

SELECT DISTINCT \* FROM(SELECT \* FROM tmp\_ps)

ORDER BY ps\_id ASC, mim\_id ASC, mim\_name ASC;

/\* In ps\_dictionary, join the two inputs of OMIM (the morbid map as ‘omim\_def’ and the PS table as ‘ps\_names’). \*/

CREATE TABLE **ps\_dictionary**(ps\_id TEXT, ps\_name TEXT, mim\_id INT, mim\_name\_morbidmap TEXT, dgp\_mim INT, entrez INT, symbol TEXT);

INSERT INTO ps\_dictionary

SELECT DISTINCT \* FROM(

SELECT l.ps\_id, r1.ps\_name, l.mim\_id,

r2.mim\_name, r2.dgp\_mim, r2.entrez\_id, r2.symbol

FROM ps l

LEFT JOIN ps\_names r1 ON l.ps\_id = r1.ps\_id

LEFT JOIN omim\_def r2 ON l.mim\_id = r2.phen\_mim

);

/\* The ps\_dictionary table, contains many D that have not been characterized in molecular terms. For instance, many D have been assigned to a certain PS but still have no DGP associated (and therefore have a key different from 3). So, delete these D. \*/

DELETE FROM ps\_dictionary WHERE dgp\_mim IS NULL;

.once c:/sqlite/LHPS/outputs/**ps\_dictionary.txt**

SELECT \* FROM ps\_dictionary;

DROP TABLE tmp\_ps; DROP TABLE ps\_names;