**/\* 3. The Phenotype Series (PS): the intersections of the PS-based LH DGP and the PC \*/**

/\* Which are the PS (with the diseases and the DGP)? Second, which are the non-null PS-PC intersections? The PS were obtained from OMIM (PS\_db.txt) and filtered (omim\_def\_ps.txt) with the same criteria used for the whole OMIM dataset (see 01\_diseases\_all.docx). \*/

.open c:/sqlite/PC/pc.db

.mode tabs

.headers ON

.import c:/sqlite/PC/inputs/PS\_def.txt tmp\_ps

CREATE TABLE **ps**(ps\_name TEXT, ps\_id TEXT, disease TEXT, mim\_id INTEGER, dgp\_mim INTEGER, entrez INTEGER, symbol TEXT);

INSERT INTO ps SELECT DISTINCT \* FROM tmp\_ps;

/\* Remove four PS that contain one DGP only. \*/

DELETE FROM ps WHERE(

ps\_id = ‘PS106210’ OR ps\_id = ‘PS145000’ OR ps\_id = ‘PS603075’ OR ps\_id = ‘PS606777’);

/\* Create PS-DGP doublets. \*/

CREATE TABLE **doublets\_ps\_dgp**(ps\_id TEXT, dgp INTEGER, symbol TEXT);

INSERT INTO doublets\_ps\_dgp

SELECT DISTINCT \* FROM(

SELECT ps\_id, entrez, symbol FROM ps);

/\* Create DGP-PC doublets \*/

CREATE TABLE **pc\_dgp**(go\_id TEXT, status TEXT, dgp INTEGER);

INSERT INTO pc\_dgp

SELECT DISTINCT \* FROM(

SELECT go\_id, status, gene\_id FROM pc\_prot);

/\* Create PS-DGP-PC triplets, by joining the two sets of doublets. \*/

CREATE TABLE **triplets**(ps\_id TEXT, dgp INTEGER, symbol TEXT, go\_id TEXT, status TEXT);

INSERT INTO triplets

SELECT DISTINCT \* FROM(

SELECT a.ps\_id, a.dgp, a.symbol, b.go\_id, b.status

FROM doublets\_ps\_dgp a

JOIN pc\_dgp b ON a.dgp = b.dgp);

DELETE FROM triplets WHERE(go\_id = ‘GO\_0043234’ OR go\_id = ‘GO\_0005667’ OR go\_id = ‘GO\_0043235’); /\* Three highly generic PC. \*/

/\* Tocreate **output\_01**, which reports all the instances whereby PSi = PSj) AND (DGPi ≠ DGPj) AND (PCi = PCj), self-join the triplets table. \*/

CREATE TABLE tmp1\_output\_01(ps\_id TEXT, dgp\_i INTEGER, dgp\_j INTEGER, go\_id TEXT, status TEXT);

INSERT INTO tmp1\_output\_01

SELECT DISTINCT \* FROM(

SELECT i.ps\_id, i.dgp AS 'dgp\_i', j.dgp AS 'dgp\_j', i.go\_id, i.status

FROM triplets i

INNER JOIN triplets j

ON (i.ps\_id = j.ps\_id) AND (i.dgp < j.dgp) AND (i.go\_id = j.go\_id)

ORDER BY i.ps\_id);

/\* To calculate the Jaccard coefficients, calculate the number of proteins in each PC (table pc\_length), the number of DGP in each PS (PS length) and the intersection of each PS/PC pair (pspc\_length). \*/

CREATE TABLE **pc\_length**(go\_id TEXT, go\_name TEXT, status TEXT, pc\_length INTEGER);

INSERT INTO pc\_length

SELECT DISTINCT \* FROM(

SELECT go\_id, go\_name, status, COUNT (DISTINCT gene\_id) FROM pc\_prot

GROUP BY go\_id, go\_name, status

ORDER BY COUNT (DISTINCT gene\_id) DESC);

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

INSERT INTO ps\_length

SELECT DISTINCT \* FROM(

SELECT ps\_id, ps\_name, count(distinct entrez) FROM ps

GROUP BY ps\_id);

CREATE TABLE **pspc\_length**(ps\_id TEXT, go\_id TEXT, dgp\_count INTEGER);

INSERT INTO pspc\_length

SELECT DISTINCT \* FROM(

SELECT ps\_id, go\_id, COUNT(DISTINCT dgp) FROM triplets

GROUP BY ps\_id, go\_id);

CREATE TABLE **output\_01**(ps\_id TEXT, dgp\_i INTEGER, dgp\_j INTEGER, go\_id TEXT, status TEXT, ps\_name TEXT, dgp\_i\_name TEXT, dgp\_j\_name TEXT, go\_name TEXT, ps\_length REAL, go\_length REAL, pspc\_length REAL);

INSERT INTO output\_01

SELECT DISTINCT \* FROM(

SELECT a.\*, b.ps\_name, c1.symbol, c2.symbol, d.go\_name, b.ps\_length, d.pc\_length, e.dgp\_count

FROM tmp1\_output\_01 a

LEFT JOIN ps\_length b ON a.ps\_id = b.ps\_id

LEFT JOIN mim2gene c1 ON a.dgp\_i = c1.entrez\_id

LEFT JOIN mim2gene c2 ON a.dgp\_j = c2.entrez\_id

LEFT JOIN pc\_length d ON a.go\_id = d.go\_id

LEFT JOIN pspc\_length e ON (a.ps\_id = e.ps\_id AND a.go\_id = e.go\_id)

);

ALTER TABLE output\_01 ADD COLUMN jacc REAL;

UPDATE output\_01

SET jacc = round(pspc\_length/(ps\_length + go\_length – pspc\_length),3);

/\* For ease of analysis, produce a table with all the DGP that are shared by each PS-PC pair expressed as semicolon-separated symbols in the same row. \*/

CREATE TABLE *triplets\_concat*(ps\_id TEXT, go\_id TEXT, dgp INTEGER, symbol TEXT);

INSERT INTO triplets\_concat

SELECT DISTINCT \* FROM(

SELECT ps\_id, go\_id, GROUP\_CONCAT(dgp,”;”), GROUP\_CONCAT(symbol,”;”)

FROM triplets

GROUP BY ps\_id, go\_id);

CREATE TABLE **output\_01\_def**(ps\_id TEXT, dgp\_ids TEXT, go\_id TEXT, status TEXT, ps\_name TEXT, dgp\_names TEXT, go\_name TEXT, ps\_length REAL, go\_length REAL, pspc\_length REAL, jacc REAL);

INSERT INTO output\_01\_def

SELECT DISTINCT \* FROM(

SELECT a.ps\_id, b.dgp, a.go\_id, a.status, a.ps\_name, c.dgp, a.go\_name, a.ps\_length, a.go\_length, a.pspc\_length, a.jacc

FROM output\_01 a

LEFT JOIN triplets\_concat b ON (a.ps\_id = b.ps\_id AND a.go\_id = b.go\_id)

LEFT JOIN triplets\_concat c ON (a.ps\_id = c.ps\_id AND a.go\_id = c.go\_id)

);

.once c:/SQLITE/pc/outputs/output\_01\_def.txt

SELECT \* FROM output\_01\_def; /\* This output is Supplementary Table S1. \*/

DROP TABLE tmp\_ps; DROP TABLE triplets\_ concat; DROP TABLE tmp1\_output\_01;