**1. The Drug-Cytochrome-Network (DCN)**

/\* In **indiana\_table.txt**, assemble the Indiana University drug-CYP interactions Main Table from <https://drug-interactions.medicine.iu.edu/>. The table contains 359 chemical entities (351 drugs plus 8 cytochromes; CYP) and 607 drug-CYP relations, which fall under three categories (‘drug is substrate of CYP’; ‘drug inibits CYP’; ‘drug induces CYP’). Accordingly, the DCN contains 359 nodes linked by 607 (drug-<relation>-CYP) arcs. Before importing the data in SQL, make the changes described in the Excel file C:/Users/gbazzoni/Dropbox/CYP450/Indiana/drugs\_cyp.xlsx. \*/

/\* Open in SQL the **binary.db** database. Then, import the indiana\_table.txt into table ***tmp1\_dcn***. tmp1\_dcn contains 607 drug\_arc\_cyp *3*-tuples (351 unique drugs, 3 arc types and 8 unique CYP). \*/

.open C:/Users/gbazzoni/Documents/SQLITE/BINARY/binary.db

.headers ON

.mode tabs

CREATE TABLE **tmp1\_dcn**(drug TEXT, atc5 TEXT, arc INT, cyp TEXT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/indiana\_table.txt tmp1\_dcn

/\* Express the arc types as 1 (drug is substrate of CYP); 2 (drug inhibits CYP); 3 (drug induces CYP). \*/

UPDATE tmp1\_dcn SET arc = 1 WHERE arc = ‘substrate’; /\* 383 arcs \*/

UPDATE tmp1\_dcn SET arc = 2 WHERE arc = ‘inhibits’; /\* 156 arcs \*/

UPDATE tmp1\_dcn SET arc = 3 WHERE arc = ‘induces’; /\* 68 arcs \*/

UPDATE tmp1\_dcn SET drug = ‘esomeprazole’ WHERE drug = ‘esameprazole’;

/\* In ***tmp2\_dcn***, identify – by self-joining (INNER JOIN) tmp1\_dcn - the 65 drug-CYP relations in which a drug is both substrate and inhibitor (or substrate and inducer) of the same CYP. \*/

CREATE TABLE **tmp2\_dcn**(drug TEXT, atc5 TEXT, arc\_1 INT, arc\_2 INT, cyp TEXT);

INSERT INTO tmp2\_dcn

SELECT i.drug AS drug, i.atc5 AS atc5, i.arc AS arc\_1, j.arc AS arc\_2, i.cyp AS cyp

FROM tmp1\_dcn AS i

INNER JOIN tmp1\_dcn AS j

ON (i.drug = j.drug) AND (i.cyp = j.cyp) AND (i.arc = 1) AND (j.arc = 2 OR j.arc = 3);

/\* Express the bidirectional arc types as 4 (drug is substrate of and inhibits CYP; i.e. 1 and 2 becomes 4); 5 (drug is substrate of and induces CYP; i.e. 1 and 3 becomes 5). \*/

ALTER TABLE tmp2\_dcn ADD COLUMN arc INT;

UPDATE tmp2\_dcn SET arc = 4 WHERE (arc\_1 = 1 AND arc\_2 = 2); /\* 53 arcs \*/

UPDATE tmp2\_dcn SET arc = 5 WHERE (arc\_1 = 1 AND arc\_2 = 3); /\* 12 arcs \*/

/\* In **dcn**, insert all the 542 arcs, i.e., the 477 non-bidirectional arcs of tmp1\_dcn (1, 2 and 3) and the 65 bidirectional arcs of tmp2\_dcn (4 and 5). First, from the 607 arcs (in tmp1\_dcn) remove (EXCEPT) the (65\*2 = 130) double arcs (in tmp2\_dcn). Then, add (UNION) the 65 bidirectional arcs (in tmp2\_dcn). Note that dcn.txt has 65 arcs less than indiana\_table.txt (607-542 = 65), because the 130 bidirectional arcs (types 1+2 and 1+3) of indiana\_table have been merged into types 4 and 5 of dcn. \*/

CREATE TABLE **dcn** (triplet\_id INTEGER PRIMARY KEY, drug TEXT, atc5 TEXT, arc INT, cyp TEXT);

INSERT INTO dcn (drug, atc5, arc, cyp)

SELECT drug, atc5, arc, cyp FROM(

/\* The 477 non-bidirectional arcs in tmp1\_dcn \*/

SELECT drug, atc5, arc, cyp FROM(

SELECT drug, atc5, arc, cyp FROM tmp1\_dcn /\* The 607 arcs in tmp1\_dcn \*/

EXCEPT

SELECT drug, atc5, arc, cyp FROM( /\* The 65\*2 = 130 double arcs in tmp2\_dcn \*/

SELECT drug, atc5, arc\_1 AS arc, cyp FROM tmp2\_dcn

UNION

SELECT drug, atc5, arc\_2 AS arc, cyp FROM tmp2\_dcn)

)

UNION

SELECT drug, atc5, arc, cyp FROM tmp2\_dcn /\* The 65 bidirectional arcs in tmp2\_dcn \*/

);

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/dcn.txt

SELECT drug, arc, cyp FROM dcn;

/\* To assemble Figure 2 of the manuscript. \*/

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/figure\_2.txt

SELECT drug, arc, cyp FROM tmp1\_dcn;

**2. The Drug-Drug Network (DDN)**

/\* The DDN portrays all the binary interactions between pairs of drugs that interact with the same CYP. The interactions can be predicted based on the way each drug in the pair interacts with the shared CYP. Specifically, we retrieve form the DCN all the pairs of drugs (drug\_A and drug\_B, with A ≠ B) that are neighbors of CYP X. This way, we obtain ‘drug\_A>arc\_AX>cyp\_X>arc\_BX>drug\_B’ *5*-tuples. Then, we gather all the possible permutations of the two drug-CYP arcs (arc\_AX plus arc\_BX) in the two *5*-tuple of each drug-drug pair and apply a set of logical rules (in table **arc\_permutations**; see manuscript, Supplementary Table S2 and Figure 4 for details). The outcome is a prediction of the effect of drug A on the levels of drug B (in either ‘A enhances B’ or ‘A reduces B’ format). Finally, the prediction of the drug-drug effect is displayed in the DDN (specifically, in the arcs that link any pair of drug-nodes). \*/

CREATE TABLE **arc\_permutations**(arc\_permut INT, arc\_i INT, arc\_j INT, arcs\_ij INT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/arc\_permutations.txt arc\_permutations

/\* In ***tmp1\_ddn***, assemble a provisional version of the DDN, by self-joining (INNER JOIN) all the drug-arc-cyp *3*-tuples (from dcn) in which two different drugs share the same CYP (note that, in column arcs\_ij, the two arc types are merged). Then, by outer joining (JOIN) the arc\_permutations table, the arcs\_ij permutations are interpreted. There are 31,275 rows in tmp1\_ddn. \*/

CREATE TABLE **tmp1\_ddn**(triplet\_id\_i INT, drug\_i TEXT, arc\_i INT, cyp\_i TEXT, triplet\_id\_j INT, drug\_j TEXT, arc\_j INT, cyp\_j TEXT, arcs\_ij TEXT, arc\_permut INT);

INSERT INTO tmp1\_ddn

SELECT a.triplet\_id\_i, a.drug\_i, a.arc\_i, a.cyp\_i, a.triplet\_id\_j, a.drug\_j, a.arc\_j, a.cyp\_j, a.arcs\_ij, b.arc\_permut

FROM(

(SELECT i.triplet\_id AS triplet\_id\_i, i.drug AS drug\_i, i.arc AS arc\_i, i.cyp AS cyp\_i, j.triplet\_id AS triplet\_id\_j, j.drug AS drug\_j, j.arc AS arc\_j, j.cyp AS cyp\_j, i.arc||’\_’||j.arc AS arcs\_ij FROM dcn AS i

INNER JOIN dcn AS j

ON i.drug != j.drug AND i.cyp = j.cyp AND i.triplet\_id < j.triplet\_id) **AS a**

JOIN

(SELECT arcs\_ij, arc\_permut FROM arc\_permutations) **AS b**

ON a.arcs\_ij = b.arcs\_ij);

/\* In ***tmp2\_ddn***, insert the rows of tmp1\_ddn, in such a way to have the source (S) drug always followed by the target (T) drug. Note that in arc\_permut 14 and 23 both drugs are S and T. \*/

CREATE TABLE **tmp2\_ddn**(drug\_S TEXT, drug\_T TEXT, cyp TEXT, arc\_permut INT);

INSERT INTO tmp2\_ddn

SELECT drug\_S, drug\_T, cyp, arc\_permut FROM(

/\* First, the drug\_i and \_j pairs in which drug\_i is the S and drug\_j is the T. \*/

SELECT drug\_i AS drug\_S, drug\_j AS drug\_T, cyp\_i AS cyp, arc\_permut AS arc\_permut FROM tmp1\_ddn

WHERE (arc\_permut = 6 OR arc\_permut = 8 OR arc\_permut = 10 OR arc\_permut = 12 OR arc\_permut = 14 OR arc\_permut = 15 OR arc\_permut = 17 OR arc\_permut = 19 OR arc\_permut = 21 OR arc\_permut = 23)

UNION

/\* Then, the drug\_j and \_i pairs in which drug\_j is the S and drug\_i is the T. \*/

SELECT drug\_j AS drug\_S, drug\_i AS drug\_T, cyp\_i AS cyp, arc\_permut AS arc\_permut FROM tmp1\_ddn

WHERE (arc\_permut = 7 OR arc\_permut = 9 OR arc\_permut = 11 OR arc\_permut = 13 OR arc\_permut = 14 OR arc\_permut = 16 OR arc\_permut = 18 OR arc\_permut = 20 OR arc\_permut = 22 OR arc\_permut = 23)

);

/\* Convert the arc\_permut of types 7, 9, 11, 13, 16, 18, 20, 22 and 23 into ‘arc\_permut minus 1’, because S and T are now in the same order S>T as in the corresponding ‘arc\_permut – 1’ pairs. This way, arc\_permut 7 becomes 6 etc, even though arc\_permut 14 and 23 remain 14 and 23 (as both drugs are ST). \*/

UPDATE tmp2\_ddn SET arc\_permut = (arc\_permut -1)

WHERE (arc\_permut = 7 OR arc\_permut = 9 OR arc\_permut = 11 OR arc\_permut = 13 OR arc\_permut = 16 OR arc\_permut = 18 OR arc\_permut = 20 OR arc\_permut = 22);

/\* Declare the predicted outcome, i.e., whether drug\_S either enhances or reduces drug\_T, based on the type of permutation types (enhancement for types 6-to-14, reduction for types 15-to-23). Note that permutations 1-to-5 do not imply mutual effects of the two drugs, whereas the mutual effects of permutations 24 and 25 cannot be decided *a priori*. \*/

ALTER TABLE tmp2\_ddn ADD COLUMN action TEXT;

UPDATE tmp2\_ddn SET action = ‘enhances’ WHERE (arc\_permut >= 6 AND arc\_permut < 15);

UPDATE tmp2\_ddn SET action = ‘reduces’ WHERE (arc\_permut >= 15 AND arc\_permut <= 23);

/\* We have deleted 465 rows from the ddn table, because of dubious interpretation. Specifically, 11 drugs (clobazam, efavirenz, isoniazid, letermovir, modafinil, omeprazole, oritavancin, perampanel, ritonavir, rucaparib, teriflunomide) are both inhbitors and inducers of different CYP (e.g., efavirenz inhibits 1A2 but induces 3A457). Also, 73 drugs (e.g., haloperidol, imipramine, phenytoin, amitriptyline, clomipramine, diazepam …) are substrates of more than one CYP (e.g., haloperidol is substrate of both 1A2 and 3A457). Thus, pair-wise combinations of these drugs (e.g., efavirenz and haloperidol) cannot be interpreted. For instance, does efavirenz enhance haloperidol (via 1A2) or reduce it (via 3A457)? After deleting these dubious arcs from the ddn, 17,249 rows are left. \*/

/\* Identify the 236 dubious interactions. \*/

CREATE TABLE **tmp3\_ddn**(drug\_S TEXT, drug\_T TEXT, cyp1 TEXT, cyp2 TEXT);

INSERT INTO tmp3\_ddn

SELECT s.drug\_S AS **drug\_S**, t.drug\_T AS **drug\_T**, s.cyp1 AS **cyp1**, s.cyp2 AS **cyp2** /\*, s.arc1s AS arc1s, s.arc2s AS arc2s, t.arc1t AS arc1t \*/

FROM

/\* The 11 drugs that are both inhibitors and inducers of different CYPs. \*/

(SELECT i.drug AS drug\_S, i.arc AS arc1s, j.arc AS arc2s, i.cyp AS cyp1, j.cyp AS cyp2

FROM tmp1\_dcn AS i

INNER JOIN tmp1\_dcn AS j

ON i.drug = j.drug AND i.arc = 2 AND j.arc = 3 AND i.cyp != j.cyp

ORDER BY drug\_S, cyp1, cyp2) **AS s**

JOIN

/\* The 73 drugs that are substrates of more than one CYP. \*/

(SELECT i.drug AS drug\_T, i.arc AS arc1t, i.cyp AS cyp1, j.cyp AS cyp2

FROM tmp1\_dcn AS i

INNER JOIN tmp1\_dcn AS j

ON i.drug = j.drug AND i.arc = 1 AND j.arc = 1 AND i.cyp != j.cyp

ORDER BY drug\_T, cyp1, cyp2) **AS t**

ON s.cyp1 = t.cyp1 AND s.cyp2 = t.cyp2;

CREATE TABLE **tmp4\_ddn**(drug\_S TEXT, drug\_T TEXT, cyp TEXT, arc\_permut INT, action TEXT, dubious TEXT);

INSERT INTO tmp4\_ddn

SELECT l.drug\_S, l.drug\_T, l.cyp, l.arc\_permut, l.action, r.drug\_S AS dubious

FROM tmp2\_ddn AS l

LEFT JOIN tmp3\_ddn AS r ON

l.drug\_S = r.drug\_S AND l.drug\_T = r.drug\_T AND (l.cyp = r.cyp1 OR l.cyp = r.cyp2);

DELETE FROM tmp4\_ddn WHERE dubious IS NOT NULL;

/\* Note that there are 846 rows in ddn in which the S enhances (or reduces) the T, by affecting more than one CYP (e.g., amiodarone enhances aripiprazole by repressing 2D6 and 3A457). In the definitive DDN, these arcs have been merged into one arc, to simplify the read-out while preserving an important information. Actually, not only are are these arcs expression of different mechanisms, but often the same drug can have different relations with different CYP (e.g., aripiprazole is substrate of 3A457, but substrate and repressor of 2D6). Nonetheless, these triplets are interpreted in the same way in the DDN (i.e., the S either enhances or reduces t he T). In conclusion, by discarding 846 arcs from the initial 17,249 arcs, 16,403 arcs are left in the DDN. \*/

CREATE TABLE **ddn**(id INTEGER PRIMARY KEY, drug\_S TEXT, action TEXT, drug\_T TEXT, atc5\_S TEXT, atc5\_T TEXT, cyps TEXT);

INSERT INTO ddn (drug\_S, action, drug\_T, atc5\_S, atc5\_T, cyps)

SELECT DISTINCT \* FROM(

SELECT drug\_S, action, drug\_T, atc5\_S, atc5\_T, GROUP\_CONCAT(DISTINCT cyp) AS cyps FROM

(SELECT l.drug\_S, l.action, l.drug\_T, r1.atc5 AS atc5\_S, r2.atc5 AS atc5\_T, l.cyp AS cyp

FROM tmp4\_ddn AS l

LEFT JOIN tmp1\_dcn AS r1 ON l.drug\_S = r1.drug

LEFT JOIN tmp1\_dcn AS r2 ON l.drug\_T = r2.drug)

GROUP BY drug\_S, action, drug\_T, atc5\_S, atc5\_T

);

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/ddn.txt

SELECT drug\_S, action, drug\_T, cyps FROM ddn;

DROP TABLE tmp1\_dcn; DROP TABLE tmp2\_dcn;

DROP TABLE tmp1\_ddn; DROP TABLE tmp2\_ddn; DROP TABLE tmp3\_ddn; DROP TABLE tmp4\_ddn;

DROP TABLE arc\_permutations;

**2. (Appendix 2.1) Using the InterCheck database to analyze the DDN**

/\* To analyze the DDN, we examined the subset of binary interactions (within the DDN) that have clinical annotations in external databases (INTERCheck and Reposi). However, in both databases the names of the drugs are in Italian. Thus, after importing the INTERCheck dataset (***intercheck***), we link the Italian names of the drugs in INTERCheck (***tmp1\_intercheck***) with the English names of the drugs in the DCN/DDN (***drugs\_dcn***), using either name identity (when available) or shared 5th level ATC codes (***drugs\_atc\_ita***) as ‘Rosetta stone’. \*/

/\* In ***intercheck***, import the whole INTERCheck dataset, which contains about 41,000 drug pairs and the clinical relevance (risk class A through D) of their association. \*/

CREATE TABLE **intercheck**(drug\_1 TEXT, drug\_2 TEXT, relevance TEXT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/InterCheck.txt intercheck

DELETE FROM intercheck WHERE drug\_1 = ‘SODIO VALPROATO’ AND drug\_2 = ‘ISONIAZIDE’;

/\* In ***tmp1\_intercheck***, assemble the list of 1389 drugs – names in Italian – from INTERCheck. \*/

CREATE TABLE **tmp1\_intercheck**(intercheck\_name TEXT);

INSERT INTO tmp1\_intercheck

SELECT drug\_1 AS intercheck\_name FROM intercheck

UNION

SELECT drug\_2 AS intercheck\_name FROM intercheck;

ALTER TABLE tmp1\_intercheck ADD COLUMN fuzzy; /\* To allow comparisons based on name likeness \*/

UPDATE tmp1\_intercheck SET fuzzy = intercheck\_name||’%’;

/\* In ***drugs\_dcn***, gather the 351 drugs – names in English - of the DCN (and DDN). \*/

CREATE TABLE **drugs\_dcn**(drug\_id INTEGER PRIMARY KEY, drug\_name TEXT, atc5 TEXT);

INSERT INTO drugs\_dcn(drug\_name, atc5)

SELECT DISTINCT drug AS drug\_name, atc5 FROM dcn ORDER BY drug\_name;

/\* In ***drugs\_atc\_ita***, assemble a list of drugs – names in Italian – that have an ATC code. To this aim, import the table ATC\_Italian.txt, which derives from the ‘ATC e principi attivi.xlsx’ file(provided by Mauro Tettamanti as of July 30, 2021). \*/

CREATE TABLE **drugs\_atc\_ita**(drug\_name\_ita TEXT, atc\_code TEXT);

.import c:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/ATC\_Italian.txt drugs\_atc\_ita

/\* Define the level of the ATC codes (based on the length of the string: the level is 1, 2, 3, 4 and 5, if the length is 1, 3, 4, 5 and 7, respectively). \*/

ALTER TABLE drugs\_atc\_ita ADD COLUMN level INT;

UPDATE drugs\_atc\_ita SET level =

(CASE WHEN length(atc\_code) = 1 THEN 1 ELSE

(CASE WHEN length(atc\_code) BETWEEN 3 AND 5 THEN (length(atc\_code)-1) ELSE

(CASE WHEN length(atc\_code) = 7 THEN 5 ELSE 0 END) END) END);

/\* Delete the rows containing the ‘/’ string, as they refer to cocktail drugs (with > 1 active principle). \*/

DELETE FROM drugs\_atc\_ita WHERE drug\_name\_ita LIKE ‘%/%’;

/\* Correct few wrong codes. \*/

UPDATE drugs\_atc\_ita SET atc\_code = ‘R06AA09’ WHERE drug\_name\_ita = ‘DOXILAMINA SUCCINATO’ AND level = 5;

UPDATE drugs\_atc\_ita SET atc\_code = ‘R06AA52’ WHERE drug\_name\_ita = ‘DIMENIDRINATO’ AND level = 5;

/\* In ***tmp2\_intercheck***, link INTERCheck drugs (tmp1\_intercheck) and ATC codes (drugs\_atc\_ita). \*/

CREATE TABLE **tmp2\_intercheck**(intercheck\_name TEXT, atc\_name\_ita TEXT, atc5 TEXT);

INSERT INTO tmp2\_intercheck

SELECT DISTINCT \* FROM(

SELECT l.intercheck\_name, r.drug\_name\_ita AS atc\_name\_ita, r.atc\_code AS atc5

FROM tmp1\_intercheck **AS l**

LEFT JOIN drugs\_atc\_ita **AS r** ON r.level = 5 AND

(

/\* 1st condition: name identity InterCheck(Ita)/ATC(Ita) \*/

(l.intercheck\_name = r.drug\_name\_ita) OR

/\* 2nd condition: likeness of name; e.g., ‘Ampicillina’ and ‘Ampicillina sodica’ \*/

(l.intercheck\_name != r.drug\_name\_ita AND r.drug\_name\_ita LIKE l.fuzzy)

));

/\* In ***tmp3\_intercheck***, translate (Italian-English) the INTERcheck drugs that are also listed in the DCN/DDN, by joining tmp2\_intercheck with drugs\_dcn (based on either name or ATC code identity). \*/

CREATE TABLE **tmp3\_intercheck**(intercheck\_name TEXT, drug\_name TEXT, atc5 TEXT);

INSERT INTO tmp3\_intercheck

SELECT DISTINCT \* FROM(

SELECT l.intercheck\_name, r.drug\_name, r.atc5 FROM

(SELECT intercheck\_name, atc5 FROM tmp2\_intercheck) **AS l**

LEFT JOIN

(SELECT UPPER(drug\_name) AS drug\_name, atc5 FROM drugs\_dcn) **AS r**

ON

/\* Italian name = English name; e.g., Amiodarone (ITA) = Amiodarone (ENG). \*/

(l.intercheck\_name = r.drug\_name) OR

/\* Italian name ≠ English name but same ATC code; e.g. Amlodipina (ITA), Amlodipine (ENG) \*/

(l.intercheck\_name != r.drug\_name AND l.atc5 = r.atc5)

);

UPDATE tmp3\_intercheck SET drug\_name = ‘ALFENTANIL’ WHERE intercheck\_name = ‘ALFENTANILE’;

UPDATE tmp3\_intercheck SET drug\_name = ‘ARTEMISININ’ WHERE intercheck\_name = ‘ARTEMISIA’;

UPDATE tmp3\_intercheck SET drug\_name = ‘COCAINE’ WHERE intercheck\_name = ‘COCAINA’;

UPDATE tmp3\_intercheck SET drug\_name = ‘DESIPRAMINE’ WHERE intercheck\_name = ‘DESIPRAMINA’;

UPDATE tmp3\_intercheck SET drug\_name = ‘ETHANOL’ WHERE intercheck\_name = ‘ALCOOL\_ETILICO’;

UPDATE tmp3\_intercheck SET drug\_name = ‘GRAPEFRUIT\_JUICE’ WHERE intercheck\_name = ‘SUCCO\_DI\_POMPELMO’;

UPDATE tmp3\_intercheck SET drug\_name = ‘PHENOBARBITAL’ WHERE intercheck\_name = ‘FENOBARBITALE’;

UPDATE tmp3\_intercheck SET drug\_name = ‘SULFAMETHOXAZOLE’ WHERE intercheck\_name = ‘SULFAMETOXAZOLO’;

UPDATE tmp3\_intercheck SET drug\_name = ‘TOBACCO’ WHERE intercheck\_name = ‘FUMO\_DI\_SIGARETTA’;

UPDATE tmp3\_intercheck SET drug\_name = ‘WORT’ WHERE intercheck\_name = ‘IPERICO’;

/\* In ***tmp4\_intercheck***, join intercheck with tmp3\_intercheck to apply the INTERCheck classes to the drug-drug interactions that involve the CYP drugs of the DCN/DDN. \*/

CREATE TABLE **tmp4\_intercheck**(drug\_A TEXT, drug\_B TEXT, class TEXT, atc5\_A TEXT, atc5\_B TEXT);

INSERT INTO tmp4\_intercheck

SELECT DISTINCT \* FROM(

SELECT LOWER(drug\_A), LOWER(drug\_B), class, atc5\_A, atc5\_B FROM(

SELECT l.drug\_1, l.drug\_2, r1.drug\_name AS drug\_A, r2.drug\_name AS drug\_B, l.relevance AS class, r1.atc5 AS atc5\_A, r2.atc5 AS atc5\_B

FROM intercheck **AS l**

JOIN tmp3\_intercheck **AS r1**

ON (l.drug\_1 = r1.intercheck\_name) AND r1.drug\_name IS NOT NULL

JOIN tmp3\_intercheck **AS r2**

ON (l.drug\_2 = r2.intercheck\_name) AND r2.drug\_name IS NOT NULL

));

/\* In ***tmp5\_intercheck***, identify the subset of CYP drugs (212) that are also found in INTERCheck. \*/

CREATE TABLE **tmp5\_intercheck**(cyp\_drug TEXT);

INSERT INTO tmp5\_intercheck

SELECT cyp\_drug FROM(

SELECT drug\_A AS cyp\_drug FROM tmp4\_intercheck

UNION

SELECT drug\_B AS cyp\_drug FROM tmp4\_intercheck);

/\* Finally, in **ddn\_intercheck**, identify the drug\_drug pairs of the DDN, for which an InterCheck definition of the clinical relevance is available in INTERCheck (2,455 out of 16,403). \*/

CREATE TABLE **ddn\_intercheck**(id INT, drug\_S TEXT, action TEXT, cyp TEXT, drug\_T TEXT, class TEXT, atc5\_S TEXT, atc5\_T TEXT);

INSERT INTO ddn\_intercheck

SELECT DISTINCT \* FROM(

SELECT l.id, l.drug\_S, l.action, l.cyps AS cyp, l.drug\_T, r.class, l.atc5\_S, l.atc5\_T

FROM ddn **AS l**

/\* k1 and k2 limit the search to the S\_ and T\_drugs that are annotated in INTERCheck. \*/

JOIN tmp5\_intercheck **AS k1** ON l.drug\_S = k1.cyp\_drug

JOIN tmp5\_intercheck **AS k2** ON l.drug\_T = k2.cyp\_drug

LEFT JOIN tmp4\_intercheck **AS r** ON

(l.drug\_S = r.drug\_A AND l.drug\_T = r.drug\_B)

OR

(l.drug\_S = r.drug\_B AND l.drug\_T = r.drug\_A)

);

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/ddn\_intercheck.txt

SELECT DISTINCT \* FROM ddn\_intercheck WHERE class IS NOT NULL;

DROP TABLE tmp1\_intercheck; DROP TABLE tmp2\_intercheck; DROP TABLE tmp3\_intercheck; DROP TABLE tmp4\_intercheck;

DROP TABLE tmp5\_intercheck; DROP TABLE drugs\_atc\_ita; DROP TABLE intercheck;

**2. (Appendix 2.2) Using the REPOSI database to analyze the DDN**

/\* To further analyze the DDN, we quantified the occurrence of the binary interactions (in the DDN) in the REPOSI registry. The files 'PAS\_duplicatiREPOSI.xlsx' (renamed 'Reposi.xlsx') and 'dizionario\_pa.xlsx' (from Pasina, February 5th, 2016) were used. In Excel, unnecessary columns (Cognome, Nome, DOB, medico, IdREP) were deleted, while keeping id\_patient, drug names and visit (0 and 2 for admission to and release from hospital, respectively). The 59,604 rows relate to 5,615 unique patients and 821 unique drugs (some of which are components of 174 cocktail drugs that contain from 2 to 7 drugs). The cocktail components were parsed into seven columns ('drug1' to 'drug7'). Then, the relevant columns were saved (reposi\_ita.txt) and imported in SQL as ***tmp1\_reposi***. [Note that ‘id\_reposi’ is a cumulative identifier of patient plus visit, because the same patient can have two id, one for visit 0 and one for visit 2.] \*/

CREATE TABLE **tmp1\_reposi**(id\_reposi INT, patient INT, visit INT, drug\_1 TEXT, drug\_2 TEXT, drug\_3 TEXT, drug\_4 TEXT, drug\_5 TEXT, drug\_6 TEXT, drug\_7 TEXT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/reposi\_ita.txt tmp1\_reposi

/\* In ***tmp2\_reposi***, list one drug per row, so that each cocktail component is in a different row. \*/

CREATE TABLE **tmp2\_reposi**(id\_reposi INT, patient INT, visit INT, drug\_name TEXT);

INSERT INTO tmp2\_reposi

SELECT DISTINCT \* FROM(

SELECT id\_reposi, patient, visit, Drug\_1 AS drug\_name FROM tmp1\_reposi WHERE drug\_1 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_2 AS drug\_name FROM tmp1\_reposi WHERE drug\_2 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_3 AS drug\_name FROM tmp1\_reposi WHERE drug\_3 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_4 AS drug\_name FROM tmp1\_reposi WHERE drug\_4 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_5 AS drug\_name FROM tmp1\_reposi WHERE drug\_5 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_6 AS drug\_name FROM tmp1\_reposi WHERE drug\_6 <> ‘’

UNION

SELECT id\_reposi, patient, visit, Drug\_7 AS drug\_name FROM tmp1\_reposi WHERE drug\_7 <> ‘’

) ORDER BY id\_reposi, patient, visit, drug\_name;

/\* Import as ***tmp3\_reposi*** a table, which contains the unique drugs names (in Italian) of the drugs in the REPOSI database, the name of the parental compounds (obtained from the file dizionario\_pa.xlsx), as well as the ATC 5th level code (provided by L. Pasina). \*/.

CREATE TABLE **tmp3\_reposi**(reposi\_name TEXT, parental\_name TEXT, atc5 TEXT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/reposi\_drugs.txt tmp3\_reposi

/\* In ***tmp4\_reposi***, join tmp2\_ and tmp3\_reposi to link each drug with parental compound and ATC code. \*/

CREATE TABLE **tmp4\_reposi**(id\_reposi INT, patient INT, visit INT, drug\_name TEXT, parental\_name TEXT, atc5 TEXT);

INSERT INTO tmp4\_reposi

SELECT DISTINCT \* FROM(

SELECT l.id\_reposi, l.patient, l.visit, l.drug\_name, r.parental\_name, r.atc5

FROM tmp2\_reposi AS l

LEFT JOIN tmp3\_reposi AS r ON l.drug\_name = r.reposi\_name);

/\* In ***tmp5\_reposi***, join tmp4\_reposi with drugs\_dcn to translate (Italian to English) the CYP drugs. \*/

CREATE TABLE **tmp5\_reposi**(id\_reposi INT, patient INT, visit INT, reposi\_name TEXT, parental\_name TEXT, drug\_id INT, drug\_name TEXT, atc5 TEXT);

INSERT INTO tmp5\_reposi

SELECT DISTINCT \* FROM(

SELECT l.id\_reposi, l.patient, l.visit, l.drug\_name AS reposi\_name, l.parental\_name,

r.drug\_id, r.drug\_name, r.atc5 FROM

(SELECT id\_reposi, patient, visit, drug\_name, parental\_name, atc5 FROM tmp4\_reposi) **AS l**

LEFT JOIN

(SELECT drug\_id, UPPER(drug\_name) AS drug\_name, atc5 FROM drugs\_dcn) **AS r**

ON

(

/\* 1st criterion: English name = Italian name (compound or parental). \*/

(l.drug\_name = r.drug\_name OR l.parental\_name = r.drug\_name)

OR

/\* 2nd criterion: English code = Italian code (5th-level ATC code) \*/

(l.atc5 = r.atc5)

)

);

/\* Finally, in ***reposi\_cyp***, select the unique fields of interest (i.e., CYP drug prescriptions). Only 146 (out of the 351 CYP-related drugs present in the DDN) are found at least once in the Reposi dataset, which amounts to a 17,8% (146/821) probability of finding a CYP-related drug for each Reposi prescribed drug. [Note for later analysis of the motifs. If we are only interested in drug triplets, retain only the patients (2,799 out of 8,581) receiving at least 3 CYP-related drugs]. \*/

CREATE TABLE **reposi\_cyp**(id\_reposi INT, patient INT, visit INT, drug\_id INT, drug\_name TEXT, atc5 TEXT);

INSERT INTO reposi\_cyp

SELECT DISTINCT \* FROM(

SELECT id\_reposi, patient, visit, drug\_id, drug\_name, atc5

FROM tmp5\_reposi WHERE drug\_id IS NOT NULL

);

/\* In **ddn\_reposi**, define the actual frequency of co-prescription of drug pairs. First, in ***tmp6\_reposi***, define all the possible pairs of drugs that have been co-prescribed to the same ‘id\_reposi’ (that is, to the same patient and in the same visit). Then, in ***tmp7\_reposi***, identify the subset of CYP drugs (146) that are also found in REPOSI. Finally, in **ddn\_reposi**, identify the drug\_drug pairs of the DDN, for which a clinical record of coprescription is available in the REPOSI datset. \*/

CREATE TABLE **tmp6\_reposi**(id\_reposi INT, patient INT, visit INT, drug\_A\_id INT, drug\_B\_id, drug\_A TEXT, drug\_B TEXT);

INSERT INTO tmp6\_reposi

SELECT DISTINCT \* FROM(

SELECT i.id\_reposi, i.patient, i.visit, i.drug\_id AS drug\_A\_id, j.drug\_id AS drug\_B\_id, i.drug\_name AS drug\_A, j.drug\_name AS drug\_B

FROM reposi\_cyp AS i

INNER JOIN reposi\_cyp AS j

ON (i.id\_reposi = j.id\_reposi) AND (i.drug\_id < j.drug\_id)

);

CREATE TABLE **tmp7\_reposi**(cyp\_drug TEXT);

INSERT INTO tmp7\_reposi

SELECT DISTINCT \* FROM(SELECT LOWER(drug\_name) AS cyp\_drug FROM reposi\_cyp ORDER BY cyp\_drug);

CREATE TABLE **ddn\_reposi**(id INT, drug\_S TEXT, action TEXT, cyp TEXT, drug\_T TEXT, drug\_S\_atc5 TEXT, drug\_T\_atc5 TEXT, id\_reposi INT, patient INT, visit INT);

INSERT INTO ddn\_reposi

SELECT DISTINCT \* FROM(

SELECT l.id, l.drug\_S, l.action, l.cyps AS cyp, l.drug\_T, l.atc5\_S, l.atc5\_T,

r.id\_reposi, r.patient, r.visit

FROM ddn **AS l**

/\* k1 and k2 limit the search to the S\_ and T\_drugs that are annotated in REPOSI. \*/

JOIN tmp7\_reposi **AS k1** ON l.drug\_S = k1.cyp\_drug

JOIN tmp7\_reposi **AS k2** ON l.drug\_T = k2.cyp\_drug

LEFT JOIN (SELECT LOWER(drug\_A) AS drug\_A, LOWER(drug\_B) AS drug\_B, id\_reposi, patient, visit FROM tmp6\_reposi) **AS r** ON

(l.drug\_S = r.drug\_A AND l.drug\_T = drug\_B)

OR

(l.drug\_S = r.drug\_B AND l.drug\_T = drug\_A)

);

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/ddn\_reposi.txt

SELECT DISTINCT \* FROM ddn\_reposi;

DROP TABLE tmp1\_reposi; DROP TABLE tmp2\_reposi; DROP TABLE tmp3\_reposi;

DROP TABLE tmp4\_reposi; DROP TABLE tmp5\_reposi;

DROP TABLE tmp6\_reposi; DROP TABLE tmp7\_reposi; DROP TABLE reposi\_cyp; DROP TABLE drugs\_dcn;

**2. (Appendix 2.3) Combining InterCheck and REPOSI to analyze the DDN (Table 1)**

/\* The following script is used to calculate the intersection of the DDN with the two clinical datasets outlined above (i.e., INTERcheck and REPOSI) and to assemble Table 1 of the manuscript. The aim is to calculate how many drug-drug interactions of the DDN are also risky associations (INTERcheck) as well as actual co-prescriptions (REPOSI). \*/

/\* First, retrieve table **drug\_pharmacol**, which summarizes some pharmacological properties of the CYP-drugs, namely which drugs are prodrugs (ProD = 1), which have a narrow therapeutic range (NThR = 1), as well as what kind of adverse reaction (ade\_id) or therapeutic failure (thf\_id) are expected, as detailed in the file drug\_pharmacology.xlsx.\*/

CREATE TABLE **drug\_pharmacol**(drug\_id INT, drug\_name TEXT, NThR INT, ProD INT, atc4 TEXT, atc3 TEXT, ade TEXT, thf TEXT, atc5\_code TEXT, atc4\_code TEXT, atc3\_code TEXT);

.import C:/Users/gbazzoni/Documents/SQLITE/BINARY/inputs/drug\_pharmacol.txt drug\_pharmacol

/\* Finally, in **ddn\_intercheck\_reposi**, define the actual frequency of the co-prescription (according to REPOSI) of the drug pairs that are clinically risky (according to InterCheck). \*/

CREATE TABLE **ddn\_intercheck\_reposi**(id INT, drug\_S TEXT, action TEXT, cyp TEXT, drug\_T TEXT, ic\_class TEXT, atc5\_S TEXT, atc5\_T TEXT, atc4\_S TEXT, atc4\_T TEXT, atc3\_S TEXT, atc3\_T TEXT, NThR INT, ProD INT, ade TEXT, thf TEXT, patient INT, atc4\_S\_code TEXT, atc3\_S\_code TEXT, atc4\_T\_code TEXT, atc3\_T\_code TEXT);

INSERT INTO ddn\_intercheck\_reposi

SELECT DISTINCT \* FROM(

SELECT l.id, l.drug\_S, l.action, l.cyp, l.drug\_T, l.ic\_class,

l.atc5\_S, l.atc5\_T,

r1.atc4 AS atc4\_S, r2.atc4 AS atc4\_T, r1.atc3 AS atc3\_S, r2.atc3 AS atc3\_T,

r2.NThR, r2.ProD, r2.ade, r2.thf,

s.patient,

r1.atc4\_code AS atc4\_S\_code, r1.atc3\_code AS atc3\_S\_code,

r2.atc4\_code AS atc4\_T\_code, r2.atc3\_code AS atc3\_T\_code

FROM (SELECT id, drug\_S, action, cyp, drug\_T, class AS ic\_class, atc5\_S, atc5\_T

FROM ddn\_intercheck /\* WHERE ic\_class IS NOT NULL \*/) **AS l**

JOIN drug\_pharmacol **AS r1** ON l.drug\_S = r1.drug\_name

JOIN drug\_pharmacol **AS r2** ON l.drug\_T = r2.drug\_name

JOIN ddn\_reposi **AS s** ON l.id = s.id /\* AND s.patient IS NOT NULL \*/

);

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/ddn\_intercheck\_reposi.txt

SELECT \* FROM ddn\_intercheck\_reposi;

**2. (Appendix 2.4) SQL Queries used to assemble the tables of the manuscript**

**/\* Table 1 \*/**

/\* How many drug pairs in the DDN are reported as risky in INTERCheck? \*/

SELECT COUNT(DISTINCT id) AS **ic\_all** FROM **ddn\_intercheck**;

SELECT COUNT(DISTINCT id) AS **ic\_risky** FROM **ddn\_intercheck** WHERE class IS NOT NULL;

SELECT class, COUNT(id) AS **ic\_risky\_class** FROM **ddn\_intercheck** /\* WHERE class IS NOT NULL \*/

GROUP BY class ORDER BY class DESC;

/\* How many drug pairs in the DDN are reported as prescribed in REPOSI? \*/

SELECT COUNT(DISTINCT id) AS **rep\_all** FROM **ddn\_reposi**;

SELECT COUNT(DISTINCT id) AS **rep\_prescribed** FROM **ddn\_reposi** WHERE patient IS NOT NULL;

SELECT COUNT(DISTINCT patient) AS **rep\_patient** FROM **ddn\_reposi**;

/\* How many drug pairs in the DDN are reported as both risky in INTERCheck and prescribed in REPOSI? \*/

SELECT COUNT(DISTINCT id) AS **ic\_rep\_all** FROM **ddn\_intercheck\_reposi**;

SELECT COUNT(DISTINCT id) AS **ic\_rep\_risky\_prescribed** FROM **ddn\_intercheck\_reposi**

WHERE ic\_class IS NOT NULL AND patient IS NOT null;

SELECT ic\_class AS class, COUNT(DISTINCT id) AS **ic\_rep\_risky\_prescribed\_class**

FROM **ddn\_intercheck\_reposi** WHERE ic\_class IS NOT NULL AND patient IS NOT null

GROUP BY ic\_class ORDER BY ic\_class DESC;

SELECT COUNT(DISTINCT patient) AS **ic\_rep\_patients** FROM **ddn\_intercheck\_reposi**

WHERE ic\_class IS NOT NULL AND patient IS NOT null;

**/\* Supplementary Table S1 \*/**

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S1.txt

SELECT drug, arc, GROUP\_CONCAT(cyp) FROM dcn GROUP BY drug, arc;

**/\* Supplementary Table S3 \*/**

/\* To ddn\_intercheck\_reposi apply sequentially the four pharmacological criteria detailed in the Introduction of the manuscript. First, in the tables *tmp\_tables\_S3\_n* (n = [*1,4*]) - rank (ext\_rank), in descending order of prescription (pt\_count), the most prescribed drug-drug interactions grouped according to the 3rd level ATC code of the source drug (up to 12 groups). Then, in the tables *table\_S3\_n*, for each of the four criteria above, rank (int\_rank), in descending order of prescription, the three most prescribed interactions grouped according to both 3rd and 4th level ATC codes of both source and target drug (limiting to the interactions that have been co-prescribed to at least three patients). \*/

/\* S3\_1. ADE (ade) due to defective inactivation (action = ‘enhances’) of a target drug (ProD != 1) \*/

CREATE TABLE *tmp\_table\_S3\_1*(ext\_rank INTEGER PRIMARY KEY, atc3\_S TEXT);

INSERT INTO tmp\_table\_S3\_1 (atc3\_S)

SELECT DISTINCT atc3\_S FROM(

SELECT atc3\_S, COUNT(DISTINCT patient) AS pt\_count FROM ddn\_intercheck\_reposi

WHERE action = ‘enhances’ AND ProD != 1

GROUP BY atc3\_S, atc4\_S, atc3\_T, atc4\_T ORDER BY pt\_count DESC) LIMIT 12;

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S3\_1.txt

**SELECT l.atc3\_S, r.“atc4\_S [drugs\_S]”, r.atc3\_T, r.“atc4\_T [drugs\_T]”, r.Patients, r.ade, r.CYPS FROM**

**(**SELECT ext\_rank, atc3\_S FROM *tmp\_table\_S3\_1***) AS l**

**JOIN**

**(**SELECT \* FROM(

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, Patients, ade, CYPS,

DENSE\_RANK() OVER (PARTITION BY atc3\_S\_code ORDER BY Patients DESC) AS int\_rank FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S, atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T, atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS Patients, ade, cyp AS CYPS

FROM ddn\_intercheck\_reposi

WHERE action = ‘enhances’ AND ProD != 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, Patients DESC)

) WHERE int\_rank BETWEEN 1 AND 3 AND Patients >= 3**) AS r**

**ON l.atc3\_S = r.atc3\_S ORDER BY ext\_rank, int\_rank;**

/\* S3\_2. THF (thf) due to defective activation (action = ‘enhances’) of a target prodrug (ProD = 1) \*/

CREATE TABLE *tmp\_table\_S3\_2*(ext\_rank INTEGER PRIMARY KEY, atc3\_S TEXT);

INSERT INTO tmp\_table\_S3\_2 (atc3\_S)

SELECT DISTINCT atc3\_S FROM(

SELECT atc3\_S, COUNT(DISTINCT patient) AS pt\_count FROM ddn\_intercheck\_reposi

WHERE action = ‘enhances’ AND ProD = 1

GROUP BY atc3\_S, atc4\_S, atc3\_T, atc4\_T ORDER BY pt\_count DESC) LIMIT 12;

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S3\_2.txt

**SELECT l.atc3\_S, r.“atc4\_S [drugs\_S]”, r.atc3\_T, r.“atc4\_T [drugs\_T]”, r.Patients, r.thf, r.CYPS FROM**

**(**SELECT ext\_rank, atc3\_S FROM *tmp\_table\_S3\_2***) AS l**

**JOIN**

**(**SELECT \* FROM(

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, Patients, thf, CYPS,

DENSE\_RANK() OVER (PARTITION BY atc3\_S\_code ORDER BY Patients DESC) AS int\_rank FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S, atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T, atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS Patients, thf, cyp AS CYPS

FROM ddn\_intercheck\_reposi

WHERE action = ‘enhances’ AND ProD = 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, Patients DESC)

) WHERE int\_rank BETWEEN 1 AND 3 AND Patients >= 3**) AS r**

**ON l.atc3\_S = r.atc3\_S ORDER BY ext\_rank, int\_rank;**

/\* S3\_3. THF (thf) due to excessive inactivation (action = ‘reduces’) of a target drug (ProD != 1) \*/

CREATE TABLE *tmp\_table\_S3\_3*(ext\_rank INTEGER PRIMARY KEY, atc3\_S TEXT);

INSERT INTO tmp\_table\_S3\_3 (atc3\_S)

SELECT DISTINCT atc3\_S FROM(

SELECT atc3\_S, COUNT(DISTINCT patient) AS pt\_count FROM ddn\_intercheck\_reposi

WHERE action = ‘reduces’ AND ProD != 1

GROUP BY atc3\_S, atc4\_S, atc3\_T, atc4\_T ORDER BY pt\_count DESC) LIMIT 12;

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S3\_3.txt

**SELECT l.atc3\_S, r.“atc4\_S [drugs\_S]”, r.atc3\_T, r.“atc4\_T [drugs\_T]”, r.Patients, r.thf, r.CYPS FROM**

**(**SELECT ext\_rank, atc3\_S FROM *tmp\_table\_S3\_3***) AS l**

**JOIN**

**(**SELECT \* FROM(

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, Patients, thf, CYPS,

DENSE\_RANK() OVER (PARTITION BY atc3\_S\_code ORDER BY Patients DESC) AS int\_rank FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S, atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T, atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS Patients, thf, cyp AS CYPS

FROM ddn\_intercheck\_reposi

WHERE action = ‘reduces’ AND ProD != 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, Patients DESC)

) WHERE int\_rank BETWEEN 1 AND 3 AND Patients >= 3**) AS r**

**ON l.atc3\_S = r.atc3\_S ORDER BY ext\_rank, int\_rank;**

/\* S3\_4. ADE (ade) due to excessive activation (action = ‘reduces’) of a target prodrug (ProD = 1) \*/

CREATE TABLE *tmp\_table\_S3\_4*(ext\_rank INTEGER PRIMARY KEY, atc3\_S TEXT);

INSERT INTO tmp\_table\_S3\_4 (atc3\_S)

SELECT DISTINCT atc3\_S FROM(

SELECT atc3\_S, COUNT(DISTINCT patient) AS pt\_count FROM ddn\_intercheck\_reposi

WHERE action = ‘reduces’ AND ProD = 1

GROUP BY atc3\_S, atc4\_S, atc3\_T, atc4\_T ORDER BY pt\_count DESC) LIMIT 12;

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S3\_4.txt

**SELECT l.atc3\_S, r.“atc4\_S [drugs\_S]”, r.atc3\_T, r.“atc4\_T [drugs\_T]”, r.Patients, r.ade, r.CYPS FROM**

**(**SELECT ext\_rank, atc3\_S FROM *tmp\_table\_S3\_4***) AS l**

**JOIN**

**(**SELECT \* FROM(

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, Patients, ade, CYPS,

DENSE\_RANK() OVER (PARTITION BY atc3\_S\_code ORDER BY Patients DESC) AS int\_rank FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S, atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T, atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS Patients, ade, cyp AS CYPS

FROM ddn\_intercheck\_reposi

WHERE action = ‘reduces’ AND ProD = 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, Patients DESC)

) WHERE int\_rank BETWEEN 1 AND 3 AND Patients >= 3**) AS r**

**ON l.atc3\_S = r.atc3\_S ORDER BY ext\_rank, int\_rank;**

DROP TABLE *tmp\_table\_S3\_1*; DROP TABLE *tmp\_table\_S3\_2*;

DROP TABLE *tmp\_table\_S3\_3*; DROP TABLE *tmp\_table\_S3\_4*;

**/\* Supplementary Table S4 \*/**

/\* To identify the contraindicated associations (INTERcheck class D), to the ddn\_intercheck\_reposi table apply sequentially the four pharmacological criteria detailed in the Introduction. Specifically, in the tables *table\_S4\_n*, for each of the four criteria, identify, in descending order of prescription, the most prescribed (class ‘D’) interactions grouped according to both 3rd and 4th level ATC codes of both source and target drug. Limit the search to the interactions that have been co-prescribed to at least one patient (or five patients for the first group, which is the most common). \*/

/\* S4\_1. ADE (ade) due to defective inactivation (action = ‘enhances’) of a target drug (ProD != 1) \*/

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S4\_1.txt

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, pt\_count, ade, “CYP(s)” FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S,

atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T,

atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T||” (“||ic\_class||”)”)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS pt\_count, ade,

cyp AS “CYP(s)”

FROM ddn\_intercheck\_reposi

WHERE ic\_class = ‘D’ AND action = ‘enhances’ AND ProD != 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, pt\_count DESC)

WHERE pt\_count >= 5;

/\* S4\_2. THF (thf) due to defective activation (action = ‘enhances’) of a target prodrug (ProD = 1) \*/

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S4\_2.txt

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, pt\_count, thf, “CYP(s)” FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S,

atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T,

atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T||” (“||ic\_class||”)”)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS pt\_count, thf,

cyp AS “CYP(s)”

FROM ddn\_intercheck\_reposi

WHERE ic\_class = ‘D’ AND action = ‘enhances’ AND ProD = 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, pt\_count DESC)

WHERE pt\_count >= 1;

/\* S4\_3. THF (thf) due to excessive inactivation (action = ‘reduces’) of a target drug (ProD != 1) \*/

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S4\_3.txt

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, pt\_count, thf, “CYP(s)” FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S,

atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T,

atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T||” (“||ic\_class||”)”)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS pt\_count, thf,

cyp AS “CYP(s)”

FROM ddn\_intercheck\_reposi

WHERE ic\_class = ‘D’ AND action = ‘reduces’ AND ProD != 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, pt\_count DESC)

WHERE pt\_count >= 1;

/\* S4\_4. ADE (ade) due to excessive activation (action = ‘reduces’) of a target prodrug (ProD = 1) \*/

.once C:/Users/gbazzoni/Documents/SQLITE/BINARY/outputs/table\_S4\_4.txt

SELECT atc3\_S, “atc4\_S [drugs\_S]”, atc3\_T, “atc4\_T [drugs\_T]”, pt\_count, ade, “CYP(s)” FROM(

SELECT atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code,

atc3\_S,

atc4\_S||“ [“||GROUP\_CONCAT(DISTINCT drug\_S)||”]” AS “atc4\_S [drugs\_S]”,

atc3\_T,

atc4\_T||“ [“||GROUP\_CONCAT(DISTINCT drug\_T||” (“||ic\_class||”)”)||”]” AS “atc4\_T [drugs\_T]”,

COUNT(DISTINCT patient) AS pt\_count, ade,

cyp AS “CYP(s)”

FROM ddn\_intercheck\_reposi

WHERE ic\_class = ‘D’ AND action = ‘reduces’ AND ProD = 1

GROUP BY atc3\_S\_code, atc4\_S\_code, atc3\_T\_code, atc4\_T\_code

ORDER BY atc3\_S\_code, atc3\_T\_code, pt\_count DESC)

WHERE pt\_count >= 1;