**Utilizing RxNorm to Support Practical Computing Applications: Capturing Medication History in Live Electronic Health Records**

Casey C. Bennetta,b

aDept. of Informatics

Centerstone Research Institute

Nashville, TN, USA

b*School of Informatics and Computing*

*Indiana University*

*Bloomington, IN, USA*

**Appendix**

**SQL (UMLS):**

-- Step 1, get ingredient and dose counters

create table umls.med\_list\_temp1 as

Select x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty, x.ingr\_cnt,

count(distinct string\_to\_array(a.atv, ' ')) as dose\_cnt

from

(

Select distinct x.aui as med\_aui, x.str as med\_name, y.str as common\_form, y.cui, y.aui, y.tty,

coalesce(rr.aui1, y.aui) as aui\_generic, y.ingr\_cnt

from umls.concept x, umls.relationship r,

(Select yy.cui, yy.aui, yy.str, yy.tty, count(rr.aui1) as ingr\_cnt from umls.concept yy

 left outer join umls.relationship rr on yy.aui=rr.aui2 and rr.rela='tradename\_of'

 where yy.sab='RXNORM' and yy.tty in ('SBDC', 'SCDC') --this subquery is mainly for counting up the duplicate ingredients in some OTC brand name drugs,

 group by yy.cui, yy.aui, yy.str, yy.tty) y --so we can filter out later

left outer join umls.relationship rr on y.aui=rr.aui2 and rr.rela='tradename\_of'

where y.aui=r.aui2

and r.rela='has\_ingredient'

and x.aui=r.aui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.metaui

group by x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty, x.ingr\_cnt

order by dose\_cnt, ingr\_cnt

;

--Step 2, Get generic counters

create table umls.med\_list\_temp2 as

Select x.med\_name, x.cui, x.aui, x.tty, count(distinct aui\_generic) as generic\_cnt, x.ingr\_cnt

from

(

Select distinct x.str as med\_name, x.cui, x.aui, x.tty, x.ingr\_cnt,

(case when x.tty='IN' then x.aui else (case when x.ingr\_cnt>1 then null else r.aui1 end) end) as aui\_generic

from (Select distinct xx.str, xx.cui, xx.aui, xx.tty, xx.sab, yy.ingr\_cnt

 from umls.concept xx left outer join umls.med\_list\_temp1 yy on yy.med\_aui=xx.aui) x

left outer join umls.relationship r on x.aui=r.aui2 and r.rela='tradename\_of'

where

x.sab='RXNORM'

and x.tty in ('IN', 'BN')

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

group by x.med\_name, x.cui, x.aui, x.tty, x.ingr\_cnt

;

--Step 3, Pull Med\_List

Select x.str as med\_name, x.cui, x.aui, x.tty,

max(case when x.tty='IN' then x.aui else (case when x.ingr\_cnt>1 or x.generic\_cnt>1 then null else r.aui1 end) end) as aui\_generic

from (Select distinct xx.str, xx.cui, xx.aui, xx.tty, xx.sab, yy.ingr\_cnt, yy.generic\_cnt

 from umls.concept xx left outer join umls.med\_list\_temp2 yy on yy.aui=xx.aui) x

left outer join umls.relationship r on x.aui=r.aui2 and r.rela='tradename\_of'

where

x.sab='RXNORM'

and x.tty in ('IN', 'BN')

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

group by x.str, x.cui, x.aui, x.tty

order by med\_name

;

--Step 4, Pull Med\_List\_Common

Select x.med\_aui, x.med\_name, x.common\_form, --This final outer query just re-orders columns and selects dose components out of array

(case when x.test[1] like '%:%' or x.test[1] like '%-%'

 or lower(x.test[1]) in ('albicans', 'million', 'mentagrophytes', 'leaf', 'antigen', 'extract') then null

 else cast(x.test[1] as numeric) end) as dose\_amt,

(case when x.test[2] is null

 or x.test[1] like '%:%' or x.test[1] like '%-%'

 or lower(x.test[1]) in ('albicans', 'million', 'mentagrophytes', 'leaf', 'antigen', 'extract') then 'Other Units'

 else x.test[2] end) as dose\_units,

x.cui, x.aui, x.tty,

x.ingr\_cnt, x.dose\_cnt

from

(

Select distinct x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty,

string\_to\_array(a.atv, ' ') as test, x.ingr\_cnt, x.dose\_cnt

from

(

Select distinct x.aui as med\_aui, x.str as med\_name, (case when y.ingr\_cnt>1 then x.str else y.common\_form end) as common\_form, y.cui, y.aui, y.tty,

(case when y.dose\_cnt>1 or y.ingr\_cnt>1 then null else coalesce(rr.aui1, y.aui) end) as aui\_generic,

y.ingr\_cnt, y.dose\_cnt

from umls.concept x, umls.relationship r,

umls.med\_list\_temp1 y left outer join umls.relationship rr on y.aui=rr.aui2 and rr.rela='tradename\_of'

where y.aui=r.aui2

and r.rela='has\_ingredient'

and x.aui=r.aui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.metaui

) x

order by med\_name, common\_form, dose\_units, dose\_amt

;

--Step 5, Pull Med\_list\_dose

Select distinct x.med\_aui, x.med\_name, --This final outer query just re-orders columns and selects dose components out of array

(case when x.test[2] is null then 'Other Units' else x.test[2] end) as dose\_units

from

(

Select distinct x.med\_aui, x.med\_name,

string\_to\_array(a.atv, ' ') as test

from

(

Select distinct x.aui as med\_aui, x.str as med\_name, (case when y.ingr\_cnt>1 then x.str else y.common\_form end) as common\_form, y.cui, y.aui, y.tty,

(case when y.dose\_cnt>1 then null else coalesce(rr.aui1, y.aui) end) as aui\_generic,

y.ingr\_cnt, y.dose\_cnt

from umls.concept x, umls.relationship r,

umls.med\_list\_temp1 y left outer join umls.relationship rr on y.aui=rr.aui2 and rr.rela='tradename\_of'

where y.aui=r.aui2

and r.rela='has\_ingredient'

and x.aui=r.aui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.metaui

) x

order by med\_name, dose\_units

;

--Clean up

drop table umls.med\_list\_temp1;

drop table umls.med\_list\_temp2;

**SQL (Monthly/Weekly RxNorm Files):**

-- Step 1, get ingredient and dose counters

create table umls.med\_list\_temp1 as

Select x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty, x.ingr\_cnt,

count(distinct string\_to\_array(a.atv, ' ')) as dose\_cnt

from

(

Select distinct x.rxaui as med\_aui, x.str as med\_name, y.str as common\_form, y.rxcui as cui, y.rxaui as aui, y.tty,

coalesce(rr.rxaui1, y.rxaui) as aui\_generic, y.ingr\_cnt

from umls.rx\_concept x, umls.rx\_relationship r,

(Select yy.rxcui, yy.rxaui, yy.str, yy.tty, count(rr.rxaui1) as ingr\_cnt from umls.rx\_concept yy

 left outer join umls.rx\_relationship rr on yy.rxaui=rr.rxaui2 and rr.rela='tradename\_of'

 where yy.sab='RXNORM' and yy.tty in ('SBDC', 'SCDC') --this subquery is mainly for counting up the duplicate ingredients in some OTC brand name drugs,

 group by yy.rxcui, yy.rxaui, yy.str, yy.tty) y --so we can filter out later

left outer join umls.rx\_relationship rr on y.rxaui=rr.rxaui2 and rr.rela='tradename\_of'

where y.rxaui=r.rxaui2

and r.rela='has\_ingredient'

and x.rxaui=r.rxaui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.rx\_attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.rxaui

group by x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty, x.ingr\_cnt

order by dose\_cnt, ingr\_cnt

;

--Step 2, Get generic counters

create table umls.med\_list\_temp2 as

Select x.med\_name, x.cui, x.aui, x.tty, count(distinct aui\_generic) as generic\_cnt, x.ingr\_cnt

from

(

Select distinct x.str as med\_name, x.cui, x.aui, x.tty, x.ingr\_cnt,

(case when x.tty='IN' then x.aui else (case when x.ingr\_cnt>1 then null else r.rxaui1 end) end) as aui\_generic

from (Select distinct xx.str, xx.rxcui as cui, xx.rxaui as aui, xx.tty, xx.sab, yy.ingr\_cnt

 from umls.rx\_concept xx left outer join umls.med\_list\_temp1 yy on yy.med\_aui=xx.rxaui) x

left outer join umls.rx\_relationship r on x.aui=r.rxaui2 and r.rela='tradename\_of'

where

x.sab='RXNORM'

and x.tty in ('IN', 'BN')

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

group by x.med\_name, x.cui, x.aui, x.tty, x.ingr\_cnt

;

--Step 3, Pull Med\_List

Select x.str as med\_name, x.cui, x.aui, x.tty,

max(case when x.tty='IN' then x.aui else (case when x.ingr\_cnt>1 or x.generic\_cnt>1 then null else r.rxaui1 end) end) as aui\_generic

from (Select distinct xx.str, xx.rxcui as cui, xx.rxaui as aui, xx.tty, xx.sab, yy.ingr\_cnt, yy.generic\_cnt

 from umls.rx\_concept xx left outer join umls.med\_list\_temp2 yy on yy.aui=xx.rxaui) x

left outer join umls.rx\_relationship r on x.cui=r.rxaui2 and r.rela='tradename\_of'

where

x.sab='RXNORM'

and x.tty in ('IN', 'BN')

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

group by x.str, x.cui, x.aui, x.tty

order by med\_name

;

--Step 4, Pull Med\_List\_Common

Select x.med\_aui, x.med\_name, x.common\_form, --This final outer query just re-orders columns and selects dose components out of array

(case when x.test[1] like '%:%' or x.test[1] like '%-%'

 or lower(x.test[1]) in ('albicans', 'million', 'mentagrophytes', 'leaf', 'antigen', 'extract') then null

 else cast(x.test[1] as numeric) end) as dose\_amt,

(case when x.test[2] is null

 or x.test[1] like '%:%' or x.test[1] like '%-%'

 or lower(x.test[1]) in ('albicans', 'million', 'mentagrophytes', 'leaf', 'antigen', 'extract') then 'Other Units'

 else x.test[2] end) as dose\_units,

x.cui, x.aui, x.tty,

x.ingr\_cnt, x.dose\_cnt

from

(

Select distinct x.med\_aui, x.med\_name, x.common\_form, x.cui, x.aui, x.tty,

string\_to\_array(a.atv, ' ') as test, x.ingr\_cnt, x.dose\_cnt

from

(

Select distinct x.rxaui as med\_aui, x.str as med\_name, (case when y.ingr\_cnt>1 then x.str else y.common\_form end) as common\_form, y.cui, y.aui, y.tty,

(case when y.dose\_cnt>1 or y.ingr\_cnt>1 then null else coalesce(rr.rxaui1, y.aui) end) as aui\_generic,

y.ingr\_cnt, y.dose\_cnt

from umls.rx\_concept x, umls.rx\_relationship r,

umls.med\_list\_temp1 y left outer join umls.rx\_relationship rr on y.aui=rr.rxaui2 and rr.rela='tradename\_of'

where y.aui=r.rxaui2

and r.rela='has\_ingredient'

and x.rxaui=r.rxaui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.metaui

) x

order by med\_name, common\_form, dose\_units, dose\_amt

;

--Step 5, Pull Med\_list\_dose

Select distinct x.med\_aui, x.med\_name, --This final outer query just re-orders columns and selects dose components out of array

(case when x.test[2] is null then 'Other Units' else x.test[2] end) as dose\_units

from

(

Select distinct x.med\_aui, x.med\_name,

string\_to\_array(a.atv, ' ') as test

from

(

Select distinct x.rxaui as med\_aui, x.str as med\_name, (case when y.ingr\_cnt>1 then x.str else y.common\_form end) as common\_form, y.cui, y.aui, y.tty,

(case when y.dose\_cnt>1 then null else coalesce(rr.rxaui1, y.aui) end) as aui\_generic,

y.ingr\_cnt, y.dose\_cnt

from umls.rx\_concept x, umls.rx\_relationship r,

umls.med\_list\_temp1 y left outer join umls.rx\_relationship rr on y.aui=rr.rxaui2 and rr.rela='tradename\_of'

where y.aui=r.rxaui2

and r.rela='has\_ingredient'

and x.rxaui=r.rxaui1

and lower(x.str) not like '%brand of%'

and lower(x.str) not like '% pet %'

) x

left outer join umls.attribute a

on a.sab='RXNORM' and a.atn='RXN\_STRENGTH' and x.aui\_generic=a.metaui

) x

order by med\_name, dose\_units

;

--Clean up

drop table umls.med\_list\_temp1;

drop table umls.med\_list\_temp2;

**DDL (Oracle):**

--Med\_List

ALTER TABLE MED\_LIST

 DROP PRIMARY KEY CASCADE;

DROP TABLE MED\_LIST CASCADE CONSTRAINTS;

CREATE TABLE MED\_LIST

(

 MED\_LIST\_ID INTEGER NOT NULL,

 MED\_NAME VARCHAR2(200 CHAR) NOT NULL,

 CUI VARCHAR2(12 CHAR) NOT NULL,

 AUI VARCHAR2(12 CHAR) NOT NULL,

 TTY VARCHAR2(20 CHAR) NOT NULL,

 AUI\_GENERIC VARCHAR2(12 CHAR),

 CREATED\_DATE DATE,

 CHANGED\_DATE DATE

)

TABLESPACE MEDIUM

PCTUSED 0

PCTFREE 10

INITRANS 1

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

LOGGING

NOCOMPRESS

NOCACHE

NOPARALLEL

MONITORING;

COMMENT ON TABLE MED\_LIST IS 'List of Medications from RxNorm';

COMMENT ON COLUMN MED\_LIST.MED\_LIST\_ID IS 'Internally generated autonumber';

COMMENT ON COLUMN MED\_LIST.MED\_NAME IS 'Name of Medication';

COMMENT ON COLUMN MED\_LIST.CUI IS 'UMLS concept atom identifier, may or may not be unique';

COMMENT ON COLUMN MED\_LIST.AUI IS 'UMLS unique atom identifier';

COMMENT ON COLUMN MED\_LIST.TTY IS 'Term Type, from UMLS, either IN=Ingredient (i.e. generic name) or

BN=Brand Name (aka Trade Name)';

COMMENT ON COLUMN MED\_LIST.AUI\_GENERIC IS 'AUI for generic med of this med, maps back to the same

table (i.e. recursive), can be null, typically only applies when tty=''BN''';

CREATE UNIQUE INDEX MED\_LIST\_PK ON MED\_LIST

(MED\_LIST\_ID)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

CREATE UNIQUE INDEX MED\_LIST\_U01 ON MED\_LIST

(AUI)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

ALTER TABLE MED\_LIST ADD (

 CONSTRAINT MED\_LIST\_PK

 PRIMARY KEY

 (MED\_LIST\_ID)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

ALTER TABLE MED\_LIST ADD (

 CONSTRAINT MED\_LIST\_U01

 UNIQUE (AUI)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

--Med\_List\_Common

ALTER TABLE MED\_LIST\_COMMON

 DROP PRIMARY KEY CASCADE;

DROP TABLE MED\_LIST\_COMMON CASCADE CONSTRAINTS;

CREATE TABLE MED\_LIST\_COMMON

(

 MED\_LIST\_COMMON\_ID INTEGER NOT NULL,

 MED\_LIST\_ID INTEGER NOT NULL,

 COMMON\_FORM VARCHAR2(300 CHAR) NOT NULL,

 DOSE\_AMT NUMBER,

 DOSE\_UNITS VARCHAR2(50 CHAR),

 CUI VARCHAR2(12 CHAR) NOT NULL,

 AUI VARCHAR2(12 CHAR) NOT NULL,

 TTY VARCHAR2(20 CHAR) NOT NULL

)

TABLESPACE MEDIUM

PCTUSED 0

PCTFREE 10

INITRANS 1

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

LOGGING

NOCOMPRESS

NOCACHE

NOPARALLEL

MONITORING;

COMMENT ON TABLE MED\_LIST\_COMMON IS 'Common Dose Forms of various meds, not all meds are included here (because some have no common dose forms in RxNorm), so outer join to main med\_list table to get all';

COMMENT ON COLUMN MED\_LIST\_COMMON.MED\_LIST\_COMMON\_ID IS 'Autonumber';

COMMENT ON COLUMN MED\_LIST\_COMMON.MED\_LIST\_ID IS 'fk to med\_list';

COMMENT ON COLUMN MED\_LIST\_COMMON.COMMON\_FORM IS 'Common Dose Form from RxNorm';

COMMENT ON COLUMN MED\_LIST\_COMMON.DOSE\_AMT IS 'Dose Amount, can be null if not existent in RxNorm';

COMMENT ON COLUMN MED\_LIST\_COMMON.DOSE\_UNITS IS 'Dose Units, can be null if not existent in RxNorm';

COMMENT ON COLUMN MED\_LIST\_COMMON.CUI IS 'UMLS concept identifier, may or may not be unique';

COMMENT ON COLUMN MED\_LIST\_COMMON.AUI IS 'UMLS unqiue atom identifier';

COMMENT ON COLUMN MED\_LIST\_COMMON.TTY IS 'Term Type';

CREATE UNIQUE INDEX MED\_LIST\_COMMON\_PK ON MED\_LIST\_COMMON

(MED\_LIST\_COMMON\_ID)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

CREATE UNIQUE INDEX MED\_LIST\_COMMON\_U01 ON MED\_LIST\_COMMON

(AUI)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

CREATE INDEX MED\_LIST\_COMM\_\_MED\_LIST\_ID\_IDX ON MED\_LIST\_COMMON

(MED\_LIST\_ID)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

ALTER TABLE MED\_LIST\_COMMON ADD (

 CONSTRAINT MED\_LIST\_COMMON\_PK

 PRIMARY KEY

 (MED\_LIST\_COMMON\_ID)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

ALTER TABLE MED\_LIST\_COMMON ADD (

 CONSTRAINT MED\_LIST\_COMMON\_U01

 UNIQUE (AUI)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

ALTER TABLE MED\_LIST\_COMMON ADD (

 CONSTRAINT FK\_MED\_LIST\_COMMON\_\_MED\_LIST

 FOREIGN KEY (MED\_LIST\_ID)

 REFERENCES MED\_LIST (MED\_LIST\_ID)

 ON DELETE CASCADE);

--Med\_List\_Dose

ALTER TABLE MED\_LIST\_DOSE

 DROP PRIMARY KEY CASCADE;

DROP TABLE MED\_LIST\_DOSE CASCADE CONSTRAINTS;

CREATE TABLE MED\_LIST\_DOSE

(

 MED\_LIST\_DOSE\_ID INTEGER NOT NULL,

 MED\_LIST\_ID INTEGER NOT NULL,

 DOSE\_UNITS VARCHAR2(50 CHAR) NOT NULL,

 DOSE\_AMT\_MIN NUMBER,

 DOSE\_AMT\_MAX NUMBER

)

TABLESPACE MEDIUM

PCTUSED 0

PCTFREE 10

INITRANS 1

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

LOGGING

NOCOMPRESS

NOCACHE

NOPARALLEL

MONITORING;

COMMENT ON TABLE MED\_LIST\_DOSE IS 'List of possible dose units for each med';

COMMENT ON COLUMN MED\_LIST\_DOSE.MED\_LIST\_DOSE\_ID IS 'Autonumber';

COMMENT ON COLUMN MED\_LIST\_DOSE.MED\_LIST\_ID IS 'fk to med\_list';

COMMENT ON COLUMN MED\_LIST\_DOSE.DOSE\_UNITS IS 'Dose Units, from RxNorm';

COMMENT ON COLUMN MED\_LIST\_DOSE.DOSE\_AMT\_MIN IS 'Minimum dosage amount, only if common dose form exists in med\_list\_common, actual prescribed amounts be be less than this in some cases';

COMMENT ON COLUMN MED\_LIST\_DOSE.DOSE\_AMT\_MAX IS 'Maximum dosage amount, only if common dose form exists in med\_list\_common, actual prescribed amounts may exceed this in some cases';

CREATE UNIQUE INDEX MED\_LIST\_DOSE\_PK ON MED\_LIST\_DOSE

(MED\_LIST\_DOSE\_ID)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

CREATE UNIQUE INDEX MED\_LIST\_DOSE\_LOOKUP ON MED\_LIST\_DOSE

(MED\_LIST\_ID, DOSE\_UNITS)

LOGGING

TABLESPACE MEDIUM

PCTFREE 10

INITRANS 2

MAXTRANS 255

STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 BUFFER\_POOL DEFAULT

 )

NOPARALLEL;

ALTER TABLE MED\_LIST\_DOSE ADD (

 CONSTRAINT MED\_LIST\_DOSE\_PK

 PRIMARY KEY

 (MED\_LIST\_DOSE\_ID)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

ALTER TABLE MED\_LIST\_DOSE ADD (

 CONSTRAINT MED\_LIST\_DOSE\_LOOKUP

 UNIQUE (MED\_LIST\_ID, DOSE\_UNITS)

 USING INDEX

 TABLESPACE MEDIUM

 PCTFREE 10

 INITRANS 2

 MAXTRANS 255

 STORAGE (

 INITIAL 64K

 MINEXTENTS 1

 MAXEXTENTS 2147483645

 PCTINCREASE 0

 ));

ALTER TABLE MED\_LIST\_DOSE ADD (

 CONSTRAINT FK\_MED\_LIST\_DOSE\_\_MED\_LIST

 FOREIGN KEY (MED\_LIST\_ID)

 REFERENCES MED\_LIST (MED\_LIST\_ID)

 ON DELETE CASCADE);

--Sequences

CREATE SEQUENCE MED\_LIST\_SEQ MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 2 NOORDER NOCYCLE ;

CREATE SEQUENCE MED\_LIST\_COMMON\_SEQ MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 2 NOORDER NOCYCLE ;

CREATE SEQUENCE MED\_LIST\_DOSE\_SEQ MINVALUE 1 MAXVALUE 999999999999999999999999999 INCREMENT BY 1 START WITH 1 CACHE 2 NOORDER NOCYCLE ;