**HBV Reverse Transcriptase (RT) Sequences Database**

by Weijun Huang

email: whuang.kent@gmail.com

This study is intended to build and utilize a relational database of biological sequences. Mutations in hepatitis B virus reverse transcriptases (HBV RT) sequences are not rare and can cause *drug resistance*, which is one of the major obstacles to HBV therapy [Rhee 2010]. Data of variant HBV RT sequences will be available on the NCBI databases, when they have been identified and published in papers.

Relational database HBV NCBI-Hits DB is designed for collecting every variant sequence of HBV RTs that is provided in the NCBI nucleotide database. The data will be transferred from the database to the relational database HBV RT DB. The database HBV RT DB is designed to

. provide reference service with respect to HBV RT *variants*,

. support HBV-RT mutation detection and analysis on biological sequences,

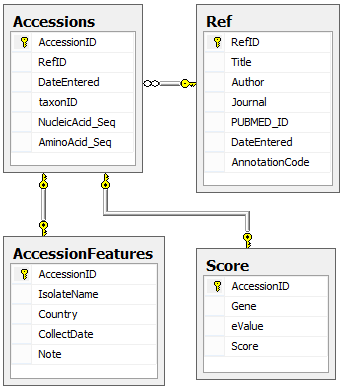
. be devoted to the quantitative study on how variant a HBV RT could be.

**Methods for the database**

1) Definition of HBV NCBI-Hits DB

This database is intended to provide reference and biology information of each published HBV RT variant.

1.1) E-R diagram:



1.2) DDL:

create table Ref

( RefID int identity(1,1) primary key not null,

Title varchar(300),

Author varchar(25),

Journal varchar(100),

PUBMED\_ID varchar(10),

DateEntered date,

AnnotationCode varchar(35)

)

create table Accessions

( AccessionID varchar(10) primary key not null,

RefID int,

DateEntered date,

taxonID varchar(6),

NucleicAcid\_Seq varchar(1032),

AminoAcid\_Seq varchar(344),

FOREIGN KEY (RefID) REFERENCES Ref(RefID)

)

create table AccessionFeatures

( AccessionID varchar(10) primary key not null,

IsolateName varchar(10),

Country varchar(20),

CollectDate date,

Note varchar(200),

FOREIGN KEY (AccessionID) REFERENCES Accessions(AccessionID)

)

create table Score

( AccessionID varchar(10) primary key not null,

Gene varchar(30) not null,

eValue double precision,

Score double precision,

FOREIGN KEY (AccessionID) REFERENCES Accessions(AccessionID)

)

Each record in the first table Ref will be a journal paper that publishes one or multiple HBV RT variant sequences. The primary key RefID of Ref will be the foreign key of the table Accessions, while each record of Accessions represents an HBV RT variant that is published in one of the papers from the table Ref. Within the NCBI nucleotide database, each sequence is assigned an accession ID. The primary key AccessionID of Accessions also serves as the primary keys and foreign keys of the tables AccessionFeatures and Score. Each row of the table AccessionFeatures represents one HBV RT variant, with data on where and when the variant sequence was collected. Each row of the table Score also present one HBV RT variant, with data on its gene names, similarity scores, etc.

2) Sequence source, retrieval and loading

1. HBV NCBI-Hits DB is seeded by the search result of HBV RTs on the NCBI nucleotide database. NCBI search results are downloadable in the format of *GenBank*.

2. In order to import data to the database, targeted information of published HBV RT variants is retrieved from the NCBI search result file. A published RT variant is considered to come with the PUBMED id of the paper. The programs for this step are coded in Python and provided in the appendix. The retrieved data is saved in a text file for each of the tables, and the field terminator of the data is ‘##’.

3. The HBV RT data can be imported using the following query:

bulk insert [Table Name]

from '[directory]\filename.txt'

with

( fieldterminator = '##',

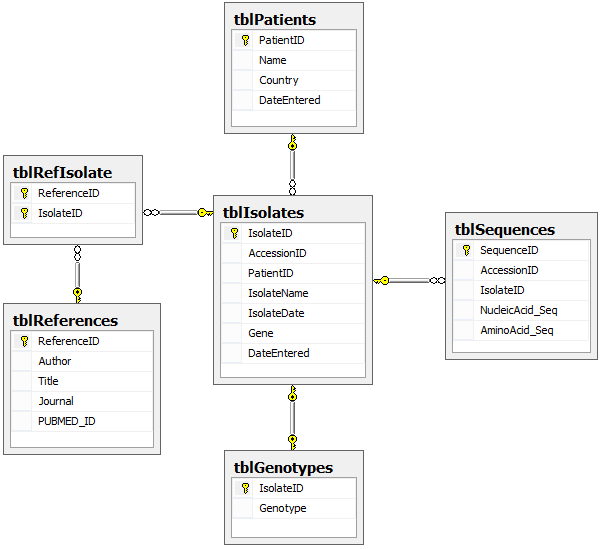
rowterminator = '\n'

)

3) Definition of HBV RT DB

This database is designed to primarily provide information on the references of HBV RT variants, and on the countries and therapies of patients from whom the HBV RT variants were isolated. The database will also provide information on the genotypes, nucleic acid sequence and amino acid sequence of the HBV RT variants.

3.1) E-R diagram:



3.1) DDL

create table tblPatients

( PatientID int identity(1, 1) PRIMARY KEY not null,

Name varchar(30),

Country varchar(20),

DateEntered date

)

create table tblIsolates

( IsolateID int identity(1, 1) PRIMARY KEY not null,

AccessionID varchar(10),

PatientID int,

IsolateName varchar(10),

IsolateDate date,

Gene varchar(30),

DateEntered date,

FOREIGN KEY (PatientID) REFERENCES tblPatients(PatientID)

)

create table tblReferences

( ReferenceID int identity(1, 1) PRIMARY KEY not null,

Author varchar(25),

Title varchar(300),

Journal varchar(100),

PUBMED\_ID varchar(10)

)

create table tblRefIsolate

( ReferenceID int not null,

IsolateID int not null,

PRIMARY KEY (ReferenceID, IsolateID),

FOREIGN KEY (ReferenceID) REFERENCES tblReferences(ReferenceID),

FOREIGN KEY (IsolateID) REFERENCES tblIsolates(IsolateID)

)

create table tblSequences

( SequenceID int identity(1, 1) primary key not null,

AccessionID varchar(10),

IsolateID int,

NucleicAcid\_Seq varchar(1032),

AminoAcid\_Seq varchar(344),

FOREIGN KEY (IsolateID) REFERENCES tblIsolates(IsolateID)

)

create table tblGenotypes

( IsolateID int primary key,

Genotype char,

FOREIGN KEY (IsolateID) REFERENCES tblIsolates(IsolateID)

)

The term “isolate” in this study is referred as to a particular HBV RT variant first-time isolated from a patient. Thus, every row in the table tblIsolates represents a HBV RT variant; in another word, it corresponds to a unique row of the table Accessions in the HBV NCBI-Hits DB. The primary key IsolateID of the table tblIsolates will be the reference key of different groups of tables (related to sequences/references/patients’ Rx history) and individual tables (virus species/RT genotypes/virus hosts).

4) Transferring data from HBV NCBI-Hits DB to HBV RT DB

1. The four attributes of table tblIsolates – IsolateName, IsolateDate, Gene and DateEntered – correspond to the attributes IsolateName, CollectDate, Gene and DateEntered of the tables AccessionFeatures, Score and Accessions in the HBV NCBI-Hits database. For each row of table tblIsolate, these attributes should share the same AccessionID value since it is the foreign key connecting the tables Accessions, AccessionFeatures, and Score.

insert into tblIsolates(AccessionID, IsolateName, IsolateDate, Gene,

DateEntered)

select Accessions.AccessionID,

AccessionFeatures.IsolateName,

AccessionFeatures.CollectDate,

Score.Gene,

Accessions.DateEntered

from [HBV NCBI-Hits].[dbo].[AccessionFeatures],

[HBV NCBI-Hits].[dbo].[Score],

[HBV NCBI-Hits].[dbo].[Accessions]

where Accessions.AccessionID = AccessionFeatures.AccessionID and

Accessions.AccessionID = Score.AccessionID

2. Each row of table tblReferences will correspond to one unique row of the table Ref in the HBV NCBI-Hits database.

insert into tblReferences(Author, Title, Journal, PUBMED\_ID)

select Ref.Author, Ref.Title, Ref.Journal, Ref.PUBMED\_ID

from [HBV NCBI-Hits].[dbo].[Ref]

3. The table tblRefIsolate is used to store the one-to-many relationship between the two tables tblReferences and tblIsolates. The attribute AccessionID is the key for this relationship.

insert into tblRefIsolate(ReferenceID, IsolateID)

select ReferenceID, IsolateID

from tblReferences,

tblIsolates,

[HBV NCBI-Hits].[dbo].[Accessions]

where Accessions.AccessionID = tblIsolates.AccessionID and

Accessions.RefID = tblReferences.ReferenceID

4. The table tblSequences is used to store the nucleic acid sequences and amino acid sequences of HBV RT variants. Of each row of this table, the values of attributes AccessionID and IsolateID are provided directly from the table tblIsolates while the sequences provided from the table Accenssions in the HBV NCBI-Hits database. Hence, data loading to the table tblSequences is feasible by joining the tables tblIsolates and Accenssions using the AccessionID attribute.

insert into tblSequences(AccessionID, IsolateID, NucleicAcid\_Seq,

AminoAcid\_Seq)

select tblIsolates.AccessionID, tblIsolates.IsolateID,

Accessions.NucleicAcid\_Seq, Accessions.AminoAcid\_Seq

from tblIsolates,

[HBV NCBI-Hits].[dbo].[Accessions]

where Accessions.AccessionID = tblIsolates.AccessionID

5. The table tblGenotypes is used to store the genotype of each isolate, and thus IsolateID is both the primary key and foreign key of the table. The genotype information is retrievable from the HBV RT search result on the NCBI nucleotide database. Item 5 of the appendix provides the Python program that was designed for retrieving genotype information. The output data file of the program can be simply bulk inserted into tblGenotypes (e.g. the bulk insert query used to load data to the HBV NCBI-Hits DB database)

Spring 2015 in CS Department

**Reference**

Rhee, S.Y. et al, 2010. Hepatitis B virus reverse transcriptase sequence variant database for sequence analysis and mutation discovery. Antiviral Research 88(3): 269-275, ELSEVIER

**Appendix**

1) Four Python programs were used to retrieve data to be imported to HBV RT NCBI-DB:

1. searchHBV\_RT\_Ref.py is used to retrieve the data of published HBV RT sequences, and extract the reference data of those sequences for the Ref table:

def searchHBV\_RT():

datafile = file('C:\Academic\Bio-CS-MTH\HBV-LiverCancer-VirusDisease\NCBI\sequence.gb')

filename = 'HBV RT sequences.txt'

newfile = open(filename, 'w')

for line in datafile:

if 'LOCUS' in line:

if '1032 bp' in line: # search for HBV RT nucleotide sequence

newfile.write(line)

for line in datafile:

newfile.write(line)

if '//\n' in line:

break

newfile.write('\n')

newfile.close()

return filename

def collectRefData(datafileName):

datafile = file(datafileName)

datafile\_1 = file(datafileName)

datafile\_2 = file(datafileName)

filename = 'HBV RT reference info.txt'

newfile = open(filename, 'w')

count\_locus = 0

count\_reference = 0

count\_l = 0

count\_r = 0

for line in datafile:

if 'LOCUS' in line:

count\_locus = count\_locus + 1

elif 'REFERENCE' in line:

count\_reference = count\_reference + 1

elif 'PUBMED' in line:

PUBMED\_ID = ''

index = 12

C = line[index]

while C != '\n':

PUBMED\_ID = PUBMED\_ID + C

index = index + 1

C = line[index] # retrieve the PUBMED id No.

while count\_r < count\_reference:

for line\_2 in datafile\_2:

if 'REFERENCE' in line\_2:

count\_r = count\_r + 1

break

for line\_2 in datafile\_2: # retrieve the first author

author = ''

index = 12

C = line\_2[index]

while C != ',' and C != '\n':

author = author + C

index = index + 1

C = line\_2[index]

break

for line\_2 in datafile\_2:

if 'TITLE' in line\_2:

title = ''

index = 12

C = line\_2[index]

while C != '\n':

title = title + C

index = index + 1

C = line\_2[index]

for line\_2 in datafile\_2:

if 'JOURNAL' not in line\_2:

title = title + ' '

index = 12

C = line\_2[12]

while C != '\n':

title = title + C

index = index + 1

C = line\_2[index]

else:

journal = ''

index = 12

C = line\_2[index]

while C != '\n':

journal = journal + C

index = index + 1

C = line\_2[index]

break

break # obtain the title and journal

while count\_l < count\_locus:

for line\_1 in datafile\_1:

if 'LOCUS' in line\_1:

count\_l = count\_l + 1

break

DateEntered = ''

index = 68

C = line\_1[index]

while C != '\n':

DateEntered = DateEntered + C

index = index + 1

C = line\_1[index] # obtain the date on which the sequence was entered

# record the reference into file

record = '##' + title + '##' + author + '##' + journal + '##' + PUBMED\_ID + '##' + DateEntered + '##' + '\n'

newfile.write(record)

newfile.close()

if \_\_name\_\_ == "\_\_main\_\_":

HBVRT\_file = searchHBV\_RT()

collectRefData(HBVRT\_file)

1. searchHBV\_RT\_Accessions.py is used to extract data of published HBV RT variants for the Accessions table, including accessions, nucleic-acid and amino-acid sequences, and to figure out the value of the foreign key RefID for each sequence:

def collectAccessionData(datafileName):

datafile = file(datafileName)

datafile\_1 = file(datafileName)

filename = 'HBV RT accession info.txt'

newfile = open(filename, 'w')

count\_locus = 0

count\_l = 0

for line in datafile:

if 'LOCUS' in line:

count\_locus = count\_locus + 1

elif 'PUBMED' in line:

PUBMED\_ID = ''

index = 12

C = line[index]

while C != '\n':

PUBMED\_ID = PUBMED\_ID + C

index = index + 1

C = line[index] # retrieve the PUBMED id No.

refFile = file('HBV RT reference info.txt')

refID = 0

for ref\_record in refFile:

refID = refID + 1

if PUBMED\_ID in ref\_record:

break # obtain the RefID in the Ref table

while count\_l < count\_locus:

for line\_1 in datafile\_1:

if 'LOCUS' in line\_1:

count\_l = count\_l + 1

break

DateEntered = ''

index = 68

C = line\_1[index]

while C != '\n':

DateEntered = DateEntered + C

index = index + 1

C = line\_1[index] # retrieve the date when the sequence was entered

for line\_1 in datafile\_1:

if 'ACCESSION' in line\_1: # obtain the accession id

accessionID = ''

index = 12

C = line\_1[index]

while C != '\n':

accessionID = accessionID + C

index = index + 1

C = line\_1[index]

for line\_1 in datafile\_1: # obtain the taxon id

if 'taxon' in line\_1:

taxonID = ''

index = 37

C = line\_1[index]

while C != '"':

taxonID = taxonID + C

index = index + 1

C = line\_1[index]

break

AminoAcid\_sequence = '' # retrieve the amino-acid sequence

for line\_1 in datafile\_1:

if 'CDS' in line\_1 and '1032' in line\_1:

for line\_1 in datafile\_1:

if 'translation=' in line\_1:

index = 35

C = line\_1[index]

while C != '\n':

AminoAcid\_sequence = AminoAcid\_sequence + C

index = index + 1

C = line\_1[index]

for i in range(0, 5):

for line\_1 in datafile\_1:

index = 21

C = line\_1[index]

while C != '\n':

AminoAcid\_sequence = AminoAcid\_sequence + C

index = index + 1

C = line\_1[index]

break

for line\_1 in datafile\_1:

index = 21

C = line\_1[index]

while C != '"':

AminoAcid\_sequence = AminoAcid\_sequence + C

index = index + 1

C = line\_1[index]

break

break

break # the amino acid sequence is fully retrieved

elif 'ORIGIN' in line\_1:

break # there is no amino acid sequence available for the HBV RT

if 'ORIGIN' in line\_1: # retrieve the nucleic-acid sequence (Case 1)

NucleicAcid\_sequence = ''

for line\_1 in datafile\_1:

if '//' not in line\_1:

index = 10

C = line\_1[index]

while C != '\n':

if C != ' ':

NucleicAcid\_sequence = NucleicAcid\_sequence + C

index = index + 1

C = line\_1[index]

else:

break # the sequence is fully retrieved

else:

for line\_1 in datafile\_1: # retrieve the nucleic-acid sequence (Case 2)

if 'ORIGIN' in line\_1:

NucleicAcid\_sequence = ''

for line\_1 in datafile\_1:

if '//' not in line\_1:

index = 10

C = line\_1[index]

while C != '\n':

if C != ' ':

NucleicAcid\_sequence = NucleicAcid\_sequence + C

index = index + 1

C = line\_1[index]

else:

break # the sequence is fully retrieved

break # stop the for loop

break # stop the for loop

# record the accession info

record = accessionID + '##' + str(refID) + '##' + DateEntered + '##' + taxonID + '##' + NucleicAcid\_sequence + '##' + AminoAcid\_sequence + '\n'

newfile.write(record)

newfile.close()

if \_\_name\_\_ == "\_\_main\_\_":

collectAccessionData('HBV RT sequences.txt')

1. searchHBV\_RT\_AccessionFeatures.py is used to extract the feature data of published HBV RT sequences for the AccessionFeatures table:

def collectAccessionFeatureData(datafileName):

datafile = file(datafileName)

datafile\_1 = file(datafileName)

filename = 'HBV RT accession features.txt'

featureFile = open(filename, 'w')

count\_locus = 0

count\_l = 0

for line in datafile:

if 'LOCUS' in line:

count\_locus = count\_locus + 1

elif 'PUBMED' in line:

while count\_l < count\_locus: # locate the RT variant in file

for line\_1 in datafile\_1:

if 'LOCUS' in line\_1:

count\_l = count\_l + 1

break

# every RT variant has a unique (LOCUS, PUBMED, ACCESSION)

for line\_1 in datafile\_1:

if 'ACCESSION' in line\_1: # obtain the accession id (from datafile\_1)

accessionID = ''

index = 12

C = line\_1[index]

while C != '\n':

accessionID = accessionID + C

index = index + 1

C = line\_1[index]

break

for line in datafile:

if 'FEATURES' in line: # obtain the isolate name, country & collection date

isolateName = '' # (from datafile)

Country = ''

CollectionDate = ''

for line in datafile:

if 'isolate=' in line:

index = 31

C = line[index]

while C != '"':

isolateName = isolateName + C

index = index + 1

C = line[index]

elif 'country=' in line:

index = 31

C = line[index]

while C != '"':

Country = Country + C

index = index + 1

C = line[index]

elif 'collection\_date=' in line:

index = 39

C = line[index]

while C != '"':

CollectionDate = CollectionDate + C

index = index + 1

C = line[index]

break

elif 'ORIGIN' in line:

break

elif '//' in line:

break

# record the accession features

record = accessionID + '##' + isolateName + '##' + Country + '##' + CollectionDate + '##' + '\n'

featureFile.write(record)

featureFile.close()

if \_\_name\_\_ == "\_\_main\_\_":

collectAccessionFeatureData('HBV RT sequences.txt')

1. searchHBV\_RT\_Score.py is used to retrieve the gene information of published HBV RT variants for the Score table:

def collectData(datafileName):

datafile = file(datafileName)

datafile\_1 = file(datafileName)

filename = 'HBV RT score info.txt'

scoreFile = open(filename, 'w')

count\_locus = 0

count\_l = 0

for line in datafile:

if 'LOCUS' in line:

count\_locus = count\_locus + 1

elif 'PUBMED' in line:

while count\_l < count\_locus:

for line\_1 in datafile\_1:

if 'LOCUS' in line\_1:

count\_l = count\_l + 1

break

for line\_1 in datafile\_1:

if 'ACCESSION' in line\_1: # obtain the accession id

accessionID = ''

index = 12

C = line\_1[index]

while C != '\n':

accessionID = accessionID + C

index = index + 1

C = line\_1[index]

for line\_1 in datafile\_1: # obtain the gene product name

if 'product' in line\_1:

Gene = ''

index = 31

C = line\_1[index]

while C != '"':

Gene = Gene + C

index = index + 1

C = line\_1[index]

break

break

# record the score info

record = accessionID + '##' + Gene + '##' + '##' + '\n'

scoreFile.write(record)

scoreFile.close()

if \_\_name\_\_ == "\_\_main\_\_":

collectData('HBV RT sequences.txt')

2) one program was used to retrieve data to be loaded to the HBV RT DB database:

5. searchHBV\_RT\_Genotypes.py is designed to retrieve the genotype information of published HBV RT variants for the tblGenotypes table:

def collectGenotypeData(datafileName):

# retrieve the genotypes of HBV RT

datafile = file(datafileName)

filename = 'HBV RT genotypes info.txt'

genotypeFile = open(filename, 'w')

IsolateID = 0

for line in datafile:

if 'PUBMED' in line:

IsolateID = IsolateID + 1

Genotype = '' # retrieve the genotype

for line in datafile:

if 'genotype' in line:

index = 0

while line[index] != 'g':

index = index + 1

Genotype = line[index + 10]

break

elif '//' in line:

break # the info of the HBV RT is fully gone over

# record the sequence

record = str(IsolateID) + '##' + Genotype + '\n'

genotypeFile.write(record)

genotypeFile.close()

if \_\_name\_\_ == "\_\_main\_\_":

collectGenotypeData('HBV RT sequences.txt')