Name – Sithalechumi Narayanan
Date – 10/14/2015
BINF 650 – Fall 2015
Homework – 6

1. Remember Kevin Bacon? This time we are going to consider the three handshake actors. In this step you will create a temporary table named co1 (co-one) with two columns. The first is the co-actors of Kevin Bacon and the second is the movies that they are in. A co-actor is in the same movie as Bacon. The first two lines of the answer are shown.

```
mysql> CREATE TEMPORARY TABLE co1 (coact1 INT, mid INT)
SELECT a1.aid as coact1, i1.mid
FROM actors a1, isin i1
WHERE a1.aid = i1.actor
AND a1.aid NOT LIKE 143
AND i1.mid IN
(SELECT i.mid FROM isin i, actors a WHERE a.aid = i.actor AND
a.firstname LIKE  'Kevin' AND a.lastname LIKE 'Bacon');
Query OK, 15 rows affected (1.62 sec)
Records: 15  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM co1 LIMIT 5;
+--------+------+
| coact1 | mid  |
+--------+------+
|      9 |    8 |
|    142 |  117 |
|    144 |  117 |
|    145 |  117 |
|    194 |  159 |
+--------+------+
5 rows in set (0.00 sec)
```

2. Create another temporary table named co2 which has the same two columns but adds 2 more columns which are all of the mid's for these co-actors and the aid of their co-actors. So, this table should have more rows than co1. The first two rows of this table are shown.

```
mysql> CREATE TEMPORARY TABLE co2 (coact1 INT, mid INT, coact2 INT, mid2
INT)
    -> SELECT coact1, start.mid, i.actor AS coact2, mid2
    -> FROM isin i, ((SELECT co1.coact1, co1.mid, isin.mid 'mid2'
    -> FROM co1, isin WHERE isin.actor = co1.coact1 AND isin.mid !=
co1.mid) AS start)
    -> WHERE i.mid = start.mid2
    -> AND i.actor != start.coact1
    -> ORDER BY coact1;
Query OK, 265 rows affected (0.01 sec)
Records: 265  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM co2 LIMIT 5;
```

```
+--------+------+--------+------+
| coact1 | mid  | coact2 | mid2 |
+--------+------+--------+------+
|      9 |    8 |    211 |  548 |
|      9 |    8 |    578 |  602 |
|      9 |    8 |    590 |  548 |
|      9 |    8 |    211 |  326 |
|      9 |    8 |     46 |  437 |
+--------+------+--------+------+
5 rows in set (0.00 sec)
```

3. Finally, create a query (not a table) that returns the 3 handshake sets for Kevin Bacon. Two example rows are shown.

After the end of query, I used **LIMIT 10**, just to get 10 records. But it was returning 1192 records.

```
SELECT coact1,(SELECT lastname FROM actors WHERE aid = coact1)
'lastname',
       co2.mid AS 'mid1', (SELECT  m.title FROM movies m WHERE m.mid =
mid1) 'title1',
       coact2, (SELECT lastname FROM actors WHERE aid = coact2)
'lastname',
       mid2, (SELECT title FROM movies m WHERE m.mid = mid2) 'title2',
       isin.actor, (SELECT lastname FROM actors WHERE aid = isin.actor)
'lastname'
FROM co2, isin
WHERE co2.mid2 = isin.mid
      AND isin.actor != co2.coact1
      AND isin.actor != co2.coact2
      AND isin.actor != (SELECT aid FROM actors WHERE firstname LIKE
'Kevin' AND lastname LIKE 'Bacon')
LIMIT 10;
+--------+----------+------+--------------------------------+--------+--
--------+------+----------------+-------+----------+
| coact1 | lastname | mid1 | title1                         | coact2 |
lastname | mid2 | title2         | actor | lastname |
+--------+----------+------+--------------------------------+--------+--
--------+------+----------------+-------+----------+
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     24 |
Perlman  |   18 | Canadian Bacon |    23 | Alda     |
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     25 |
Torn     |   18 | Canadian Bacon |    23 | Alda     |
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     26 |
Aykroyd  |   18 | Canadian Bacon |    23 | Alda     |
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     23 |
Alda     |   18 | Canadian Bacon |    24 | Perlman  |
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     25 |
Torn     |   18 | Canadian Bacon |    24 | Perlman  |
|     22 | Candy    |  339 | Planes, Trains and Automobiles |     26 |
Aykroyd  |   18 | Canadian Bacon |    24 | Perlman  |
|      22 | Candy    |   339 | Planes, Trains and Automobiles |      23 |
Alda     |   18 | Canadian Bacon |    25 | Torn     |
|      22 | Candy    |   339 | Planes, Trains and Automobiles |      24 |
Perlman  |   18 | Canadian Bacon |    25 | Torn     |
|      22 | Candy    |   339 | Planes, Trains and Automobiles |      26 |
Aykroyd  |   18 | Canadian Bacon |    25 | Torn     |
|      22 | Candy    |   339 | Planes, Trains and Automobiles |      23 |
Alda     |   18 | Canadian Bacon |    26 | Aykroyd  |
+--------+----------+------+--------------------------------+--------+--
---------+------+----------------+-------+----------+
10 rows in set (0.00 sec)
```

4. Using the GenBank tables. Create a temporary table named aasplice for splice information with the following columns: accession, pid, aaseq1 where aaseq1 is the amino acid sequence just for that splice. Two columns are shown although the sequences had to be truncated here. The genes with more than one splice will have more than one entry in this new table. The lengths of the items in the third column are equal to the splice length.

```
mysql> CREATE TEMPORARY TABLE aasplice (asid INT AUTO_INCREMENT PRIMARY
KEY,  accession VARCHAR (20), pid INT(11), aaseq1 text) SELECT
accession, pid, aaseq AS aaseq1  FROM protein;
Query OK, 85518 rows affected (0.19 sec)
Records: 85518  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM aasplice LIMIT 3;
```

```
+------+-----------+------+---------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
---------------------------------------------------------------------+
| asid | accession | pid  | aaseq1
|
+------+-----------+------+---------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
---------------------------------------------------------------------+
|    1 | AB001339  |    1 |
ARHRRLAEITEMIHTASLVHDDVVDEADLRRNVPTVNSLFDNRVAVLAGDFLFAQSSWYLANLDNLEVVKLL
SEVIRDFAEGEILQSINRFDTDTDLETYLEKSYFKTASLIANSAKAAGVLSDAPRDVCDHLYEYGKHLGLAF
QIVDDILDFTSPTEVLGKPAGSDLISGNITAPALFAMEKYPLLGKLIEREFAQAGDLEQALELVEQGDGIRR
SRELAANQAQLARQHLSVLEMSAPRESLLELVDYVLGRLH |
|    2 | AB001339  |    2 |
MGRLDQDSEGLLLLTSNGKLQHRLAHREFAHQRTYFAQVEGSPTDEDLEPLRRGITFADYPTRPAIAKIITE
PDFPPRNPPIRYRASIPTSWLSITLTEGRNRQVRRMTAAVGFPTLRLVRVQIQVTGRSPQQGKGKSAATWCL
TLEGLSPGQWRPLTPWEENFCQQLLTGNPNGPWQKKFGDRR
|
|    3 | AB001339  |    3 |
MSYLIAVVANRIAAEEAYTTLEQAGFAQKNLTIIGTGYKTADEFGLVDPKKQAIKRAKLMAIWLVPFGFAAG
YCFNLITGLSTLDWAGDPGNHIVGGLLGAIGGTMGSFFVGGGVGLSFGSGDSLPYRNLLQAGKYLVVVAGGE
LQKQRATNLLRPLNPEYLQGYTAPDEAFV
|
+------+-----------+------+------------------------------------------
---------------------------------------------------------------------
---------------------------------------------------------------------
-------------------------------------------------------------------+
3 rows in set (0.00 sec)
```

5. This question does not have to be performed in a single command. Prove that the lengths of the splices in your new table aasplice are equal to the lengths of the original proteins. For example, gene G1 has 2 splices of lengths 10 and 20. There are two entries for it in aasplice with these two sequences. The original sequence from the protein table has an aaseq of length 30. Your process will show that the two splices stored in aasplice add up to the same length as the original sequence. You must show that this is true for all proteins.

The following is a temporary table that has the pid, total DNA sequence length from splices table and protein sequence length from the temporary table aasplice.

```
mysql> CREATE TEMPORARY TABLE dnaPTNlength
    -> (pid INT(11), ProteinSeqLength INT(11), DNASeqLength INT(11))
    -> SELECT DISTINCT s.pid, LENGTH(a.aaseq1) AS 'ProteinSeqLength',
    -> s.stop-s.start+1 AS 'DNASeqLength'
    -> FROM splices s, aasplice a
    -> WHERE s.pid = a.pid;
Query OK, 85554 rows affected (4 min 48.85 sec)
Records: 85554  Duplicates: 0  Warnings: 0
```

```
mysql> SELECT * FROM dnaPTNlength LIMIT 10;
+------+------------------+--------------+
| pid  | ProteinSeqLength | DNASeqLength |
+------+------------------+--------------+
|    1 |              256 |          772 |
|    2 |              185 |          558 |
|    3 |              173 |          522 |
|    4 |              233 |          702 |
|    5 |              358 |         1077 |
|    6 |              312 |          939 |
|    7 |              362 |         1089 |
|    8 |              360 |         1083 |
|    9 |              659 |         1980 |
|   10 |              669 |         2010 |
+------+------------------+--------------+
10 rows in set (0.00 sec)
```

Now to compare, we need to multiply the protein sequence by 3, because for every amino acid there are 3 nucleotides. And also we need to subtract 3 nucleotides from the DNA sequence as we don’t need the three nucleotides that codes for stop codon.

TRIAL 1

```
mysql> SELECT ProteinSeqLength*3, DNASeqLength-3 FROM dnaPTNlength LIMIT
15;
```

```
+--------------------+----------------+
| ProteinSeqLength*3 | DNASeqLength-3 |
+--------------------+----------------+
|                768 |            769 |
|                555 |            555 |
|                519 |            519 |
|                699 |            699 |
|               1074 |           1074 |
|                936 |            936 |
|               1086 |           1086 |
|               1080 |           1080 |
|               1977 |           1977 |
|               2007 |           2007 |
|                606 |            606 |
|               1029 |           1029 |
|               1047 |           1047 |
|                804 |            804 |
|                945 |            945 |
+--------------------+----------------+
15 rows in set (0.00 sec)
```

TRIAL 2

```
mysql> SELECT ProteinSeqLength*3, DNASeqLength-3 FROM dnaPTNlength d,
aasplice a WHERE d.pid = a.pid
 AND accession = 'AE004091' LIMIT 10;
```

```
+--------------------+----------------+
| ProteinSeqLength*3 | DNASeqLength-3 |
+--------------------+----------------+
|               1542 |           1542 |
|               1101 |           1101 |
|               1107 |           1107 |
|               2418 |           2418 |
|                771 |            771 |
|                534 |            534 |
|               1704 |           1704 |
|               2052 |           2052 |
|                945 |            945 |
|                549 |            549 |
+--------------------+----------------+
10 rows in set (4.52 sec)
```

TRIAL 3

```
mysql> SELECT ProteinSeqLength*3, DNASeqLength-3 FROM dnaPTNlength d,
aasplice a
    -> WHERE d.pid = a.pid
    -> AND accession = 'AE003852' LIMIT 10;
```

```
+--------------------+----------------+
| ProteinSeqLength*3 | DNASeqLength-3 |
+--------------------+----------------+
|                165 |             165 |
|                432 |             432 |
|               1392 |            1392 |
|               1623 |            1623 |
|                255 |             255 |
|                354 |             354 |
|                135 |             135 |
|                735 |             735 |
|                669 |             669 |
|                744 |             744 |
+--------------------+-----------------+
10 rows in set (2.12 sec)
```

6. Here we are trying to find any two splices that are exactly the same. Create another table in which only those cases where there is an exact match are stored. The columns should be: id (primary key, auto_increment), asid1, asid2. Show the first 10 rows of this table and a count of the total number of entries. You should have five commands.

1. Create the empty temporary table.

```
mysql> CREATE TEMPORARY TABLE exactmatch
    -> (id INT AUTO_INCREMENT PRIMARY KEY, asid1 TEXT,
    -> asid2 TEXT)
    -> ;
Query OK, 0 rows affected (0.01 sec)
```

2. Create the procedure to perform the work.

```
mysql> DELIMITER //
mysql> CREATE  PROCEDURE mine()
    -> BEGIN
    ->   INSERT INTO exactmatch(asid1, asid2)
    ->   SELECT a.aaseq AS asid1, b.aaseq AS asid2
    ->   FROM protein a, protein b
    ->   WHERE a.pid != b.pid
    ->    AND a.aaseq = b.aaseq
    ->    AND a.pid < b.pid;
    -> END;//
Query OK, 0 rows affected (0.00 sec)
```

3. Call the procedure.

```
mysql> CALL mine();
-> //
Query OK, 4121 rows affected (7 min 18.38 sec)
```

4. Show the first 10 rows of the temporary table.

```
mysql> SELECT * FROM exactmatch LIMIT 10;
    -> //
```

```
+----+------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
--+---------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
-----------------------------------------------------------------------+
| id | asid1
| asid2
|
+----+------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
--+---------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
-----------------------------------------------------------------------+
|  1 |
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
LIDRAKKFGVRPSAVYYALKKMKINRKKKELRYRERNREERVKYYRMLRELIKLYGSQAIVYIDESGFEAIQ
ACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTGSLNAEGFEGWLKLYLLPSLDIPSILI
MDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRARMYAPIDTSLDEIIRSYCGV |
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
LIDRAKKFGVRPSAVYYALKKMKINRKKKELRYRERNREERVKYYRMLRELIKLYGSQAIVYIDESGFEAIQ
ACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTGSLNAEGFEGWLKLYLLPSLDIPSILI
MDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRARMYAPIDTSLDEIIRSYCGV |
|  2 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  3 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  4 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  5 |
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
```

```
LIDRAKKFGVRPSAVYYALKKMKINRKKKNYVIEKETGRNELSTIEC
|
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
LIDRAKKFGVRPSAVYYALKKMKINRKKKNYVIEKETGRNELSTIEC
|
|  6 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  7 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  8 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
|  9 |
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
MLRELIKLYGSQAIVYIDESGFEAIQACIYAWSKKGKKVYGDRQGKRGVRENLVAGRRKGKKDLIAPMVFTG
SLNAEGFEGWLKLYLLPSLDIPSILIMDNAPIHRKTAIKELAKEAGHEVLFLPKYSPDLNDIEHDFSALKRA
RMYAPIDTSLDEIIRSYCGV
|
| 10 |
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
LIDRAKKFGVRPSAVYYALKKMKINRKKKNYVIEKETGRNELSTIEC
|
MAYSLDLRQRVVAYIEAGGKITEASKIYKIGKASIYRWLNRVDLSPTKVERRHRKLDWEALKKDVEENPDAR
LIDRAKKFGVRPSAVYYALKKMKINRKKKNYVIEKETGRNELSTIEC
|
+----+------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
--+---------------------------------------------------------------------
------------------------------------------------------------------------
------------------------------------------------------------------------
-----------------------------------------------------------------------+
10 rows in set (0.00 sec)
```

5. Count the total number of rows in the temporary table.

```
mysql> SELECT COUNT(*) FROM exactmatch;
    -> //
+----------+
| COUNT(*) |
+----------+
|     4121 |
+----------+
1 row in set (0.00 sec)
```