Lab 2 Write Up & Discussion Questions

- Gary Miller

Discussion Questions

- What is the relation schema for each data set that describes the association between their entities? In both tables, Park and Apop, the relation schema are the same and includes an entry (named id in the tables), entry name, status, protein name, gene name, organism, and length. After actually creating a table in the database to hold this information, this can be seen by using the .schema command in SQLite3 in the terminal.
- How are the column headers helpful for designing your schemas? The column headers are helpful because anyone who is looking at this database for the first time, can easily and immediately find out which each column of data is actually describing. For this reason, it is extremely important to have simple yet descriptive naming conventions similar to that in object-oriented programming.
- What is an appropriate primary key for each data set? An appropriate primary key for this data would be the entry, which is held in the first column. This is because the entry is the genetic code of each gene and is therefore unique to each individual gene, an essential characteristic of a primary key.
- What kinds of memory (VARCHAR memory allocation) do you think you need to create your base? Why? Due to the consistent uncertainty of the length of some data entries, it is most logical to simply use a "NOT NULL" allocation of memory to prevent unexpected memory overload errors. In the event of memory constraints issues, one would have to parse the data to find instances where a memory allocation limit could be set without fear of error.

Creating the Database

The database was created by first downloading the necessary files containing information on Parkinson's's disease and Apoptosis from uniprot.org and selecting the "tab-separated" format of these files. A separate file Lab2SQL.txt contains the commands that were entered to create tables for this data and then import it into the newly created database.

Querying the Database Tables

- Write a query that will return the count of elements in the Entry columns in both tables. This query was conducted by using the count command which in general looks like:

  ```
  SELECT count(attribute) FROM table;
  ```

  Specifically with these tables the query would look like:

  ```
  SELECT count(id) FROM Park;
  SELECT count(id) FROM Apop;
  ```

  The output of these commands showed that there are 127,369 and 100,083 entries in Park and Apop respectively.

- Write a query that will return the distinct count elements in the Entry column. This query was conducted by using the distinct and count commands which in general looks like:

  ```
  SELECT count(DISTINCT attribute) FROM table;
  ```

  Specifically with these tables the query would look like:

  ```
  SELECT count(DISTINCT id) FROM Park;
  SELECT count(DISTINCT id) FROM Apop;
  ```

  The output of these commands showed that the id of entries are distinct meaning there are 127,369 and 100,083 distinct entries and total entries in Park and Apop respectively.

- Discuss: From the above two queries, is this column a good primary key for each table? why or why not? (if not, then what column would you recommend, instead?) Yes, this is an ideal column to use as a primary key, because the number of total entries and the number of distinct entries is equal in both tables.

- Write a query that will return the number of records associated with "Zea mays (Maize)" in both tables. You might want to first determine where entity is found in the table to create your query. Opposed to gleaning the hard to read and cumbersome data sheet, a search was done on Zea mays to find out it is an organism and therefore the query should look in that column of data. The general query to find the count of entries that belong to a certain column of data would look like this:

  ```
  SELECT count(attribute) FROM table WHERE attribute == "value";
  ```

  Specifically with these tables the query would look like:

  ```
  SELECT count(id) FROM Park WHERE organism == "Zea mays (Maize)";
  SELECT count(id) FROM Apop WHERE organism == "Zea mays (Maize)";
  ```

The output of these commands showed that 9 and 13 entries matched the organism "Zea mays (Maize)" in Park and Apop respectively.

- Write a query that will report how many organisms were listed in each table. In order to determine the number of organisms that were listed in each table the distinct command should be used. The general format for finding the number of distinct values in a column looks like:

  ```
  SELECT count(DISTINCT attribute) FROM table;
  ```

  Specifically with these tables the query would look like:

  ```
  SELECT count(DISTINCT organism) FROM Park;
  SELECT count(DISTINCT organism) FROM Apop;
  ```

  The output of these commands showed that 863 and 26,858 distinct organism entries were in Park and Apop respectively.

- Write a query that will return the number of organisms which are common to both tables (as in, the intersection of the tables for this attribute). Using the guidance given by the lab assignment sheet that demonstrates how to link to tables using the command structure:

  ```
  select distinct A.Entry, P.Entry from Apop A, Park P where A.Entry == P.Entry;
  ```

  The number of distinct organisms common to both tables was then found by modifying the command to query for organism opposed to Entry and also to return the count opposed to the list of distinct organisms. The modified command looked like:

  ```
  SELECT count(DISTINCT A.organism), count(DISTINCT P.organism) from Apop A, Park P where
  ```

  The output of this command showed that there are 148 distinct organisms that are common to both tables.

- Write a query that will return the number of proteins which are common to Apoptosis and Parkinson's, which are associated to the Zea mays (Maize) organism. Similar to the previous query that was used to find the distinct number of organisms that are common to both tables but changing a few attributed to find the number of proteins common to both tables that are associated with Zea mays (Maize) the following query was inputted:

  ```
  SELECT count(DISTINCT A.protein_name), count(DISTINCT P.protein_name) from Apop A, Park
  ```

  The output of the command showed that 7 and 8 distinct proteins where associated with the organism Zea mays (Maize) in Park and Apop respectively.

- Modify this query to print the first 15 proteins (Entry entities), according to Zea mays (Maize) are related. By modifying the previous query to not consider the count nor distinct entries and limit the list to only 15 entries the new query looked like:

```
SELECT A.protein_name, P.protein_name from Apop A, Park P where A.organism == "Zea mays
```

The output of this query was a long list of text that looked like: Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Mutant DNA dependent RNA polymerase D largest subunit rpd1-14 Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Mutant DNA dependent RNA polymerase D largest subunit rpd1-7 Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Mutant required to maintain repression 2 Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Purple plant 1 (Fragment) Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Required to maintain repression 2 Dolichyl-diphosphooligosaccharide–protein glycosyltransferase subunit DAD1 (Oligosaccharyl transferase subunit DAD1) (EC 2.4.99.18) (Defender against cell death 1) (DAD-1) (Fragment)|Ribosomal protein S2 (Fragment) Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|DNA-directed RNA polymerase subunit (EC 2.7.7.6) Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|Mutant DNA dependent RNA polymerase D largest subunit rpd1-14 Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|Mutant DNA dependent RNA polymerase D largest subunit rpd1-7 Anamorsin homolog (Fe-S cluster assembly protein DRE2 homolog)|Mutant required to maintain repression 2

- Create a query to determine how many genes are in common in both tables, for all organisms (i.e., the intersection of all information about

genes across all organisms). The query used to find the number of genes that are common to organisms in each table was similar to the querries used in the previous questions and looked like:

```
SELECT COUNT(DISTINCT A.organism), COUNT(DISTINCT P.organism) FROM Apop A, Park P WHERE
```

The output of this query showed that 566 and 306 genes that were similar to organisms Park and Apop resepctively.

- Create another query to determine the names of first ten of these genes which are at the intersection of both tables, across all organisms. This query was similar to the last one except the count command was removed and a limit command was added:

```
SELECT DISTINCT A.organism, P.organism FROM Apop A, Park P WHERE A.gene_name == P.gene_
```

The output of this query returned the first 15 genes that where common to organisms of both tables and returned

Daboia russelii (Russel's viper) (Vipera russelii)|Acris crepitans (northern cricket frog) Daboia russelii (Russel's viper) (Vipera russelii)|Agalychnis spurrelli (Gliding leaf frog) (Agalychnis litodryas) Daboia russelii (Russel's viper) (Vipera russelii)|Allobates trilineatus (three-striped rocket frog) Daboia russelii (Russel's viper) (Vipera russelii)|Allophryne ruthveni (Tukeit Hill frog) Daboia russelii (Russel's viper) (Vipera russelii)|Amazophrynella minuta (Tiny tree toad) (Dendrophryniscus minutus) Daboia russelii (Russel's viper) (Vipera russelii)|Anotheca spinosa (spiny-headed treefrog) Daboia russelii (Russel's viper) (Vipera russelii)|Atelopus peruensis (Peru stubfoot toad) Daboia russelii (Russel's viper) (Vipera russelii)|Atheris nitschei (Great lakes bush viper) Daboia russelii (Russel's viper) (Vipera russelii)|Atropoides olmec Daboia russelii (Russel's viper) (Vipera russelii)|Bitis arietans (African puff adder) Daboia russelii (Russel's viper) (Vipera russelii)|Bokermannohyla astartea (Paranapiacaba treefrog) Daboia russelii (Russel's viper) (Vipera russelii)|Bothriechis nigroviridis (Black-speckled palm pit viper) Daboia russelii (Russel's viper) (Vipera russelii)|Bothriechis schlegelii (Eyelash palm pitviper) Daboia russelii (Russel's viper) (Vipera russelii)|Bothrocophias microphthalmus