Solution for CIS 452/552 Assignment Four

1. *See a relational DB on a company at http://www.cs.uoregon.edu/classes/15W/cis452/assignments/en_6e_fig_3-6.pdf .*

a) If you can choose to put all or some of tables into a Key-Value database, will you gain any benefit to do so? If yes, give one example key-value pair to show your idea of implementation.

Answer:

Yes, we can put some or even all tables into a Key-Value database. We will gain benefit to do so including

1. There is no schema in the value design, which promotes the flexibility of the database.
2. It is easy to implement with parallel and concurrency systems.
3. It is easy to scale up the system.

Example:

Key = SSN,

Value = {

{

FName,

Minit,

Lname,

Bdate,

Address,

Sex,

Salary,

Super_ssn,

Dno

```
        }
        {
                Dependent_name,
                Dependent_dex,
                Dependent_Bdate,
                Dependent_relationship
        }
        {
                Work_On_Project_No,
                Work_On_Project_Hours
        }
        [{
                Work_On_Project_No_2,
                Work_On_Project_Hours
        }]
        …
}
Key = Project Number,
Value = {
                {
                        Pname,
                        Plocation,
                        DNum,
                        Dname,
                        Mgr_ssn,
```

Mgr_start_date

}

}

b) If you can choose to put all or some of tables into a Document database, will you gain any benefit to do so? If yes, give one example Document (e.g., in JSON) to show your idea of implementation.

Answer:

The benefit of using document based database are:

1. Related data are stored contiguously on disk which makes it easier to access and distributed across machines while preserving its locality.
2. No translation is needed for the objects to SQL queries, we can just stores the objects in OOP language into one document.
3. It will be easy to store unstructured data since no schema is needed in the document database.
4. It is schema free as well which promotes the flexibility.

Here is an example JSON Document of the record of John B. Smith, whose Ssn is 123456789:

```
{
  Ssn: “123456789”,
  Fname: “John”,
 Minit: “B”,
 Lname: “Smith”,
 Bdate: “1965-01-09”,
 Address: “731 Fondren, Houston, TX”,
 Sex: “M”,
 Salary: 30000,

Supervisor: {
      Ssn: 333445555,
      Fname: “Franklin”,
      Minit: “T”,
      Lname: “Wong”},
```

Department: {
Dname: “Research”,

Messenger: {
Ssn: “333445555”,
Start_date: “1988-05-22”},
Location: [“Bellaire”, “Sugarland”]}
Works_on: [{Pname: “ProductX”, Plocation: “Bellaire”, Hours: 32.5},
{Pname: “ProductY”, Plocation: “Sugarland”, Hours: 7.5}]
Dependent: [{name: “Michael”, Sex: “M”, Bdate: “1988-01-04”, Relationship: “Son”},
{name: “Alice”, Sex: “F”, Bdate: “1988-12-30”, Relationship: “Daughter”},
{name: “Elizabeth”, Sex: “F”, Bdate: “1967-05-05”, Relationship: “Spouse”}] }

c) If you can choose to put all or some of tables into a Column-Family database, will you gain any benefit to do so? If yes, give one example Column Family to show your idea of implementation.

Answer:

The advantage of Column-Family databases includes

1. It serializes all of the values of a column together on the disk storage. It can accelerate some column based DB operations
2. It also allows you to pre-compute the column based result sets and store them in a single row for efficient data retrieval. It is more robust.

The example of the implementation can be based on employees:

123456789	Fname	...	Project1...	Project2...	Dependent1...	Dependent2...	...
	John		1	2	Michael	Alice	
333445555	Fname	...	Project1...	Project2...	Project3...	Project4...	...
	Franklin		2	3	10	20	

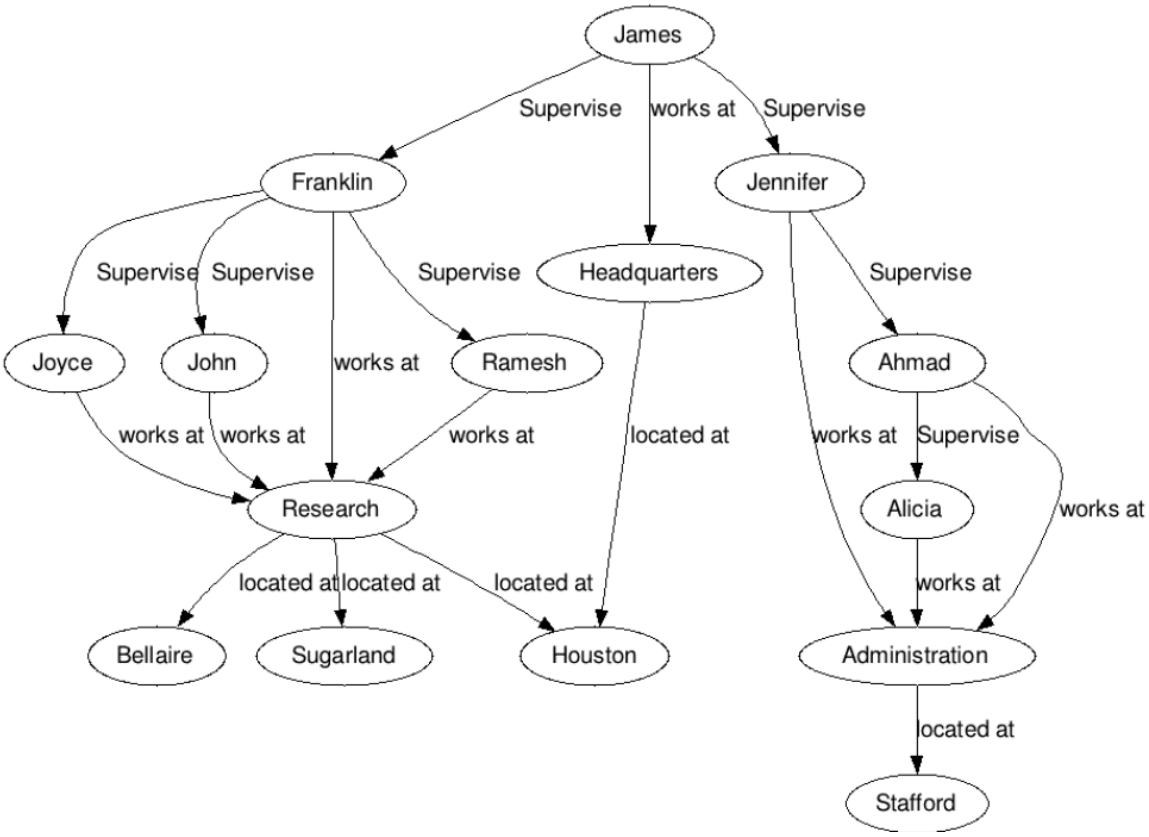
d) If you can choose to put all or some of tables into a Graph database, will you gain any benefit to do so? If yes, give a portion of the graph to show your idea of implementation.

Answer:

Graph database has the following advantages:

1. It allows different kinds of data to be stored easily.
2. Certain kinds of searches such as finding associations between different kinds of data will be quick and easy.

Example of graph database can be as follows:



2. Assume Mark created a Document database (e.g., CouchDB) for the company DB in 1). Then he wants to add more information in. Here is a possible employee document, which incorporates the works on information. A project document would be separate.

{

"fname": "Alicia",

"minit": "J",

"lname": "Zeleya",

"ssn": "999887777",

"bdate": {

"year": 1961, "month": 1, "day": 19

},

"address": "3321 Castle, Spring, TX",

"sex": "F",

"salary": 25000,

"works_on": {

"10": 10,

"30": 30

},

"superssn": "987654321",

"dno": 4

}

Based on the above new document, you may figure out roughly what is Mark's design for this document database. Please describe example Map and Reduce steps to explain how to add a map-reduce function to determine the average number of hours that an employee works on each project. The idea is that the map part should emit a pair for each works on entry (note: it must have one to be an interesting employee document). The key would be the project number and the value the number of hours worked. The reduce section will average the values for each key. You do not need to produce the project name.

Answer:

The map function: map each document to a set of pairs of project-number and working hours, e.g., emit (1, 32.5), (2,10), (1, 20)…

```
function (doc) {
        if (doc.works_on != NULL) {
        var wo = doc.work_on;
        wo.forEach(function(element)){
                emit(element[0], element[1]);
```

```
            }
        }
    }
```

The reduce function: combine the working hours for each project: (1, 52.5), (2, 37.5) … and divide each total hours by counts of employees (i.e., counts of pairs for the same project number). The results will be (1, 26.25) (2, 12.5)….

```
function(projectNo, hours){
    return [projectNo, sum(hours)/employee.count];
}
```

3. Explain the Map step and Reduce step for the following relational DB operations.

a) Projection.

Answer:

Map: Map/emit each data tuple with projected attributes.

Reduce: Do nothing

b) Selection.

Answer:

Map: Map/emit each data tuple if selection condition is satisfied.

Reduce: Do nothing

c) Group by.

Answer:

Map: Map/emit each data tuple with group by attribute as key.

Reduce: Aggregate the data tuple according to the aggregation function

d) Sort-merge join.

Answer:

Case one: Map-side join

Map: Map pair of joinable (same value on the joined attributes) data tuples to a single tuple with data from both instances of the pair.

Reduce: Do nothing.

Case two: Reduce-side join

Map: Map all tuples and emit tuple sets with specified join key

Reduce: Reduce tuples from each set, which share the same key.

4. Suppose that the data mining task is to cluster the following nine points (with (x,y) representing location) into three clusters: A1(3,10), A2(3,8), A3(9,3), B1(5,8), B2(7,5), B3(6,4), C1(2,2), C2(5,9), C3(6,7) Suppose initially we assign A1, B1 and C1 as the center of each cluster, respectively. Please add a Map-reduce function for the K-means algorithm. Show the results for the first two iterations and explain how Map-reduce can help.

Answer:

Map Reduce:

Let K_1, K_2, K_3 be the three centroids for the current iteration, let d(P, K) be the distance between points P and K.

Map: We map each points with coordinates i, j, P(i, j) to l if P is closest to centroid K_l. A sample map function could be

```
map(P) {
        emit(index_of_cloest_cetroid(K1, K2, K3, P), P)
}
```

Reduce:

```
reduce(tuples){
        return [tuples.centroid_index, sum(tuples.x)/tuples.count,
sum(tuples.y)/tuples.count];
}
```

Execution:

Iteration 1:

Cluster 1: A_1, A_2 (or without A_2 because the distance A_2 to B_1 is the same)
Centroid 1: (3, 9)

Cluster 2: C_2, B_1, C_3, B_3, B_2, A_3 (or with A_2)
Centroid 2: (19/3, 6)

Cluster 3: C_1
Centroid 3: (2, 2)

Iteration 2:

Cluster 1: A_1, A_2, B_1, C_2 (or without B_1)
Centroid 1: (4, 35/4)

Cluster 3: A_3, B_2, B_3, C_3, (or with B_1)
Centroid 1: (7, 19/4)

Cluster 3: C_1
Centroid 3: (2, 2)

Benefit:

The map reduce can help in the sense that all these operations can run in parallel.

5. A database has six transactions. Let min sup = 50%.

TID	items_sold
T001	A, B, C, D, E, F
T002	B, O, S, C, W, T
T003	A, U, T, F, W, D
T004	O, A, B, C, F, X
T005	O, A, C, D, F, Y
T006	B, C, D, E, W, Z

Please add a Map-reduce function for the Apriori algorithm to generate all frequent itemsets. Show the results for each step and explain how Map-reduce can help.

Answer:

Map: For each iteration, generate frequent items.

```
function(doc) {
    var iteration;
    var frequent_itemset;
    if(0 == iteration)
        for_each(trans = doc.transactions)
            for_each(item = trans.items)
                frequent_itemset.add(item);
    else{
        // the function below take each two frequent_itemset, generate the union of them and add in the set.
        frequent_itemset = set( Cartesian_union(frequent_itemset) );
    }
    for_each(trans in doc.transactions){
        for_each(item in frequent_itemset){
            if(trans.contains(item) ){
                emit(item, 1);
            }
        }
    }
}
```

Reduce: Count all the emitted items, compare with the support threshold.

```
reduce(keys, values, reducer){
    var count = _count;
    if(count / doc.transactions.length > doc.support)
        return (c, value)
}
```

L_1 (frequency >= 3):

A, 4

B, 4
C, 5
D, 4
F, 4
O, 3
W, 3

Candidate C_2

AB, AC, AF, AD, AO, AW, BC, BD, BF, BO, BW, CD, CF, CO, CW, DF, DO, DW, FO, FW, OW.

L_2

A,C 4
A,D 4
A,F 4
B,C 4
C,D 3
C,F 3
C,O 3
D,F 3

C_3:

ACD, ACF, ADF, CDF

L_3

ACF 3
ADF 3

C_4

None.

Benefit:
The map reduce function can help process the map reduce function in parallel.