# *Assignment No. 1*

## *WSU ID: 011716089*

**1.** Give one example of Big Data application you know. Use the detailed example to explain each of the five Big V's. If you are required to design a database system for this application, what are the best data models (relational, XML, RDF, among others) you would use to represent the data and why?

One of the real-world examples of Big Data application is Health Care industry. In past due to confined progression in innovation, healthcare frameworks utilization of big data faltered. The major cause for this postponed progress is conflicting and solidified data. But presently expository changes in Big Data has upgraded the healthcare by controlling personalized symptom signs and prescription analysis. Big Data can perform on much more cellular levels like medicinal formulation patterns breakdown as well as give a profound understanding of suitable prescriptions.

Due to expansive applications of eHealth and smart health wearable innovation, the volume of information has expanded at an exponential rate. Information like Heart rate checking, number of steps and calorie check can be mapped and analyzed. Such highlights of Big Data can deliver results which can be advantageous and productive. A few of these benefits incorporate cost reduction since there's less chances of having to perform pointless diagnosis, anticipating outbreak of epidemics and early detection of diseases. As a result of enormous amount of data, it is categorized using five V's: Velocity, Volume, Value, Variety and Veracity.

### ***Velocity:***

Critical and quick updates are imperative for emergency situations in healthcare. In such circumstances the system's reaction time is a fundamental figure and additionally can lead to competitive advertise to endeavor for advancement.

The issues with such request come about in abuse of resources and slower execution rate. A dependable arrangement would be characterizing the information source for only required and essential attributes yielding quality reports and enhancement.

### ***Volume:***

There's no question that big data is gigantic. A report from EMC states that overall information volume in 2013 sums up to 4.4 zettabytes of data and is anticipated to develop aggressively. Since the data is transitory, appropriate labeling and curated investigation on Big Data can turn out to be a boon to restorative health care framework. Unlike platforms like Netflix, binge or amazon, healthcare information can be exceptionally beneficial. Data such as genome sequence, sensory feedback, information claims when associated in inventive ways, can deliver modern revelations.

To handle such enormous information, modern storage procedures ought to be executed. They must moreover guarantee that their framework can keep up with the following V on the list without abating down basic capacities like EHR access or communications.

***Value:***

The main objective of analytics is to recover some value from information. These values can decide the results delineating the usefulness and productivity of data. Since of the size and complexity of information, determining value from analytics begins by characterizing particular use cases to handle.

Big Data can offer assistance in different ways such as therapeutic income, prediction, precipitation proposal, illnesses discovery and remedy. Numerous organizations are endeavoring to achieve these objectives and producing significant insights that can be applied to real-world issues may be a complicated and challenging. These values comply to vigorous data administration, Internet of Things, qualified data analyst and pave the way to imaginative approach for prorogation observations to users.

***Variety:***

In general, the results produced depends on cramming of heterogeneous data sources. Vast the variety is, superior is the quality of output.

Analyzing assortment of data may well be burdening for healthcare frameworks. Emerging issues with introducing huge variety of data can lead to information segregation. Moreover, such isolated data can't be compared for examination, limiting the insights that can learned. Engineers are working on providing a common stage to break down heterogeneous sources by designing complex API's to extend the assortment quotient.

***Veracity:***

Trust is an extremely important aspect when it comes to patient care. Veracity of a dataset cannot be easily verified, providers cannot use the insights derived from data that is incomplete, biased, or noisy. An extensive time is spent cleaning up data before it can be used, healthcare space is a region where accuracy is the primary necessity such that the risk factor remains negligible.

Data integrity and quality require extensive efforts to increase the levels of trust, it is not at all easy when large number of systems allow free text or other unstructured inputs. Data and information governance tend to be the backbones of healthcare organizations to ensure that their data is clean, complete, standardized, and utilizable.

According to me, the foremost reasonable model for Healthcare would be relational model. Relational model preferences incorporate organized framework for easy access, Multiple client accessibility which is fundamental in healthcare industry and common inquiry languages. In case of query language, most of them are optimized for organized databases which improves the execution process as well as the accessibility of critical information. Another benefit of utilizing relational model is simple and specific query data access. In case of Unstructured frameworks, recovering particular data can be a tedious assignment. In conclusion, relational model is the most suitable data model for healthcare system.

**2.** (a) Consider the following terms: relation schema, relational database schema, domain, attribute, attribute domain, relation instance. Give what these terms are with the above Airport database. Give one small (4-5 tuples) instance of the Airport table.

### *Relation schema:*

Relation schema in straightforward terms can be clarified as detailed description of attributes and their relation which is the consistent definition of a table. The data that Relation schema provides are column names and related information types.

Airport ID => integer

Name => char/string

City => char/string

Country => char/string

IATA => char/string

ICAO => char/string

Latitude => double

Longitude => double

Altitude => integer

Timezone => double

DST => char/string

Tz database time zone => char/string

Type => char/string

Source => char/string

### *Relational Database schema:*

A common definition of database could be a collection of interconnected or correlated information or structure. Hence, a relational database schema is a cluster of relational states in a correlated way satisfying the constraints set. Relational database schema depicts relational linkage between columns or attributes.

The Relational Database schema for airport dataset would be

AIRPORT (Airport ID, Name, City, Country, IATA, ICAO, Latitude, Longitude, Altitude, Timezone, DST, Tz database time zone, Type, Source)

### *Domain:*

A domain indicates all conceivable values that table can store. In outline, a domain could be a set of satisfactory values that a column is permitted to contain.

Example ~ The domain of Marital Status has a set of possibilities: Married, Single, Divorced.

Airport ID = unique integer value only,

Name = only alphabetical characters(no integer value permitted)

City = only alphabetical characters(no integer value permitted)

Country = only alphabetical characters(no integer value permitted)

IATA = only combination of 3 character values permitted

ICAO = combination of characters whose length is 4

Latitude = a positive or negative decimal value,

Longitude = a positive or negative decimal value,

Altitude = a positive integer value,

Timezone = a positive or negative double value,

DST = single character value,

Tz database time zone = combination of character values,

Type = combination of character values,

Source = combination of character values

***<u>Attributes:</u>***

Attributes can be thought of as columns within a table.

Airport ID, Name, City, Country, IATA, ICAO, Latitude, Longitude, Altitude, Timezone, DST, Tz database time zone, Type, Source

***<u>Attribute domain:</u>***

Attribute domain refers to data type associated with column.

Airport ID = int,

Name = varchar (100),

City = varchar (50),

Country = varchar (50),

IATA = varchar (3),

ICAO = varchar (4),

Latitude = decimal (18, 8),

Longitude = decimal (18, 8),

Altitude = int,

Timezone = decimal (18,5),

DST = varchar (1),

Tz database time zone = varchar (50),

Type = varchar (50)

Source = varchar (50)

***Relation instance:***

A relation instance is a set of tuples known as rows or records that each acclimate to the schema of the relation.

***Tuple:***

- 507,"London Heathrow Airport","London","United Kingdom","LHR","EGLL",51.4706,-0.461941,83,0,"E","Europe/London","airport","OurAirports"
- 26,"Kugaaruk Airport","Pelly Bay","Canada","YBB","CYBB",68.534401,-89.808098,56,-7,"A","America/Edmonton","airport","OurAirports"
- 3127,"Pokhara Airport","Pokhara","Nepal","PKR","VNPK",28.200899124145508,83.98210144042969,2712,5.75,"N","Asia/Katmandu","airport","OurAirports"
- 8810,"Hamburg Hbf","Hamburg","Germany","ZMB",\N,53.552776,10.006683,30,1,"E","Europe/Berlin","station","User"12,5.75,"N","Asia/Katmandu","airport","OurAirports"

(b) There are three databases in the OpenFlight dataset: Airport, Airline, and Route. Give the schema of these three databases and mark the primary keys, foreign keys and provide examples of functional dependencies you identified over the three tables. [You may draw a diagram to illustrate the schema, PKs, FKs and FDs]

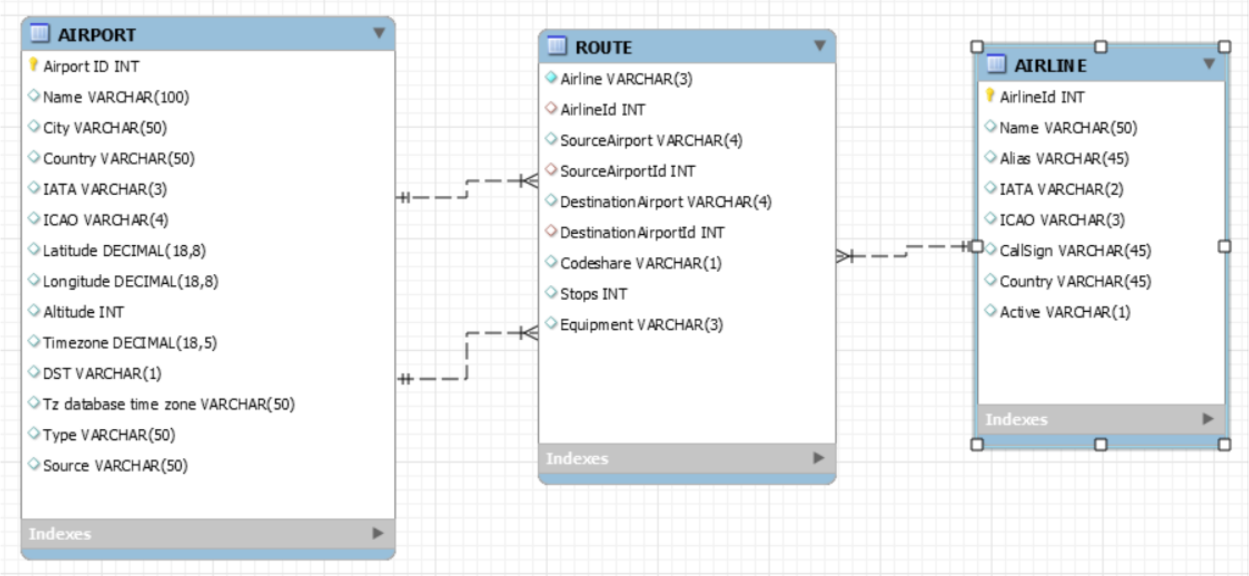
Data schema for three databases is as follows:

Airport: Primary Key = AirportId

Airline: Primary Key = AirlineId

Route:

- Foreign Key = (AirlineId) source Airline
- Foreign key = (SourceAirportId) source Airport
- Foreign key = (DestinationAirportId) source Airport



AIRPORT
Airport ID INT
Name VARCHAR(100)
City VARCHAR(50)
Country VARCHAR(50)
IATA VARCHAR(3)
ICAO VARCHAR(4)
Latitude DECIMAL(18,8)
Longitude DECIMAL(18,8)
Altitude INT
Timezone DECIMAL(18,5)
DST VARCHAR(1)
Tz database time zone VARCHAR(50)
Type VARCHAR(50)
Source VARCHAR(50)
Indexes

ROUTE
Airline VARCHAR(3)
AirlineId INT
SourceAirport VARCHAR(4)
SourceAirportId INT
DestinationAirport VARCHAR(4)
DestinationAirportId INT
Codeshare VARCHAR(1)
Stops INT
Equipment VARCHAR(3)
Indexes

AIRLINE
AirlineId INT
Name VARCHAR(50)
Alias VARCHAR(45)
IATA VARCHAR(2)
ICAO VARCHAR(3)
CallSign VARCHAR(45)
Country VARCHAR(45)
Active VARCHAR(1)
Indexes

The schema diagram portrays the functional dependency of three databases. There's one to many relationship between Airport and route schema. It can clarified by occasions where an air terminal having multiple diverse routes. Besides, comparative relation is observed between Airline and route schema.

Following are examples of functional dependencies in this database:

- Airport Table:

AirportId → Name

Name → IATA, ICAO

AirportId → Latitude, Longitude

- Airline Table:

AirlineId → Name

Name → Alias

Name → IATA, ICAO

Name → CallSign

Name may not be treated as a unique entity depending on any particular instance.

• Route Table:

SourceAirportId, DestinationAirportId → Airline

SourceAirportId, DestinationAirportId → CodeShare

SourceAirportId, Equipment → Airline

Since within the Route table we don't have any single key that can distinguish unique entities, so to do that we need to form a composite key on column basis.

c. Recall Armstrong's axioms.

1. Reflexivity rule: if Y ⊆ X then X → Y

2. Augmentation rule: if X → Y then XZ → YZ

3. Transitivity rule: if X → Y and Y → Z then X → Z

1. Give two examples for using Armstrong's inference rules to induce new FDs from the set of FDs you designed in question 2 (b).

Example:

- AirportId → Name, and Name → IATA then AirportId → IATA
- AirlineId → Name, and Name → Alias then AirlineId → Alias

(2) Prove the following inference rules also hold, using FD definition and Armstrong's Axioms.

a. decomposition rule: if X → YZ then: X → Y and X → Z

AirportId → IATA, ICAO

AirportId → IATA

AirportId → ICAO

It proves the Decomposition rule of Armstrong's Axioms such that if X → YZ then: X → Y and X → Z.

b. Pseudo transitivity: if X → Y and YW → Z then: XW → Z

AirlineId → Name

Name, IATA → ICAO

AirlineId, IATA → ICAO

This proves the pseudo transitivity rule of Armstrong's axiom if X → Y and YW → Z then: XW → Z.

(d) Given a relation R(A1, A2, A3, A4), with three FDs A2, A3 → A4 ; A3, A4 → A1; A1, A2→ A3. Provide the 3NF and BCNF form of the schema and explain why.

Relation (A1, A2, A3, A4)

FDs:

A2, A3 -> A4

A3, A4 -> A1

A1, A2 -> A3

For checking 3NF:

The candidate keys are

{ A2,A1},

{ A2,A3}

For all the FDs, the LHS is super key or the RHS are all key attributes. This is true for all the FDs.

Thus the above relation is already in 3NF.

For checking BCNF:

FD A3,A4 --> A1 is non-trivial and its LHS is not a superkey. Thus, this does not follow BCNF.

For converting to BCNF:

FD [A3,A4 --> A1] violates BCNF.

Table is split:

R2 (A3,A4,A1 )

R3 (A2,A3,A4 )

To Change over this relation to BCNF we just have to make sure that within the 3NF must not exist any nontrivial functional reliance of attributes other than candidate key. The relations R2 and R3 already follow BCNF and thus the above relations are in BCNF.

**3.** Consider the following database schema:

Movies (Title, Director, Actor);

Location (Theater, Address, Phone number);

Schedule (Theater, Title, Time).

Express the following queries in relational algebra (select σ, project ∏ , Cartesian product X, join (theta-join)).

Q1: which theaters feature “Zootopia”?

$$\sigma title = "zootopia"\ \sigma_{title} = "Zootopia"^{(Schedule)}$$

Q2: List the names and address of theaters featuring a film directed by Steven Spielberg.

$$\sigma_{title} = "Steven\ Spielberg"^{(Schedule)} \bowtie \Pi_{Theater,Address}(Location)$$

Q3: What is the address and phone number of the Le Champo theater?

$$\sigma_{Address \wedge Phone\ number \wedge theater} = "Le\ Champo"^{(Location)}$$

Q4: List pairs of actors that acted in the same movie. (* you want to use renaming on Movies and join the Movies with its copy Movie’).

$$\Pi_{Actor}\ (Movies) \bowtie \Pi_{Title \wedge Actor}\ (Movies')$$

**4.** This sets of questions test the understanding of basic database search operators. Consider a join $\bowtie_{R.A = S.B}$. We ignore the cost of output the result, and measure the cost with the number of I/Os. Given the information about relations to be joined below:

Relation $S$ contains 20,000 tuples and has 10 tuples per block. Relation $R$ contains 100,000 tuples and has 10 tuples per block. Attribute $B$ is the primary key of $S$. In total, 52 blocks are available in memory. Assume neither relation has any index.

For S:

$$\boldsymbol{N_s = 20000}$$

$$B_s = \frac{N_s}{10(Given)}$$

$$B_s = \frac{20000}{10}$$

$$\boldsymbol{B_s = 2000}$$

For R:

$$\boldsymbol{N_r = 100000}$$

$$B_r = \frac{N_r}{10(Given)}$$

$$B_r = \frac{100000}{10}$$

$$\boldsymbol{B_r = 10000}$$

a. (10) Describe a block nested join algorithm, Give the cost of joining $R$ and $S$ with a block nested loops join.

In Block Nested Loop Join, when relations ‘r’ and relation ‘s’ has to be joined, the outer loop is for reading the blocks of relation ‘r’ and inner loop is reading the blocks of relation ‘s’. If relation ‘r’ and relation ‘s’ are small enough to fit into the main memory than the join operation is performed more effectively.

In Block nested loop join, before performing the join operation the relations to be joined are first placed into the main memory. In Block Nested Loop Join algorithm, the number of disk accesses consists of two operations – one is to read the blocks of relation ‘r’ and other to access the disk for reading the blocks of relation ‘s’.

```
for each block $B_R$of ‘r’ do
{blocks of relation ‘r’ are scanned one by one.
        for each block $B_S$ of s do
```

```
    {blocks of relation ‘r’ are scanned one by one.
        Compute $B_R B_S$in memory
    }
}
```

The Block Nested Loop Join algorithm is an advanced algorithm of the nested loop join algorithm which is used for transfer of blocks efficiently rather than transferring the tuples of the participating relations in the join operation. The block nested loop joins algorithm works by reading a block of tuples, from the outer and inner relation. In BNLJ (one block at a time) chunks of each relation is transferred from hard disk to main memory where join operations is performed

$$Cost = \frac{B_R}{52} * B_S + B_R \ Block\ Transfers, +2\left(\frac{B_R}{52}\right) Seeks$$

$$Cost = \frac{10000}{52} * 2000 + 10000, +2\left(\frac{10000}{52}\right)$$

$$\boldsymbol{Cost\ =\ 394615.3846\ Block\ Transfers, 384.615\ Seeks}$$

Worst Case:

= $B_R * B_S\ transfer, 2B_R\ Seeks.$

$= 10000 * 2000\ transfers, 2 * 10000\ Seeks$

$= 20000000\ transfers, 20000\ Seeks$

Best Case:

$= B_S + B_R transfers\ , 2\ seeks$

$= 2000 + 10000\ transfers, 2\ seeks$

$= 12000\ transfers, 2\ seeks$

b. (15) Describe a sort-merge join algorithm. Give the cost of joining $R$ and $S$ with a sort-merge join.

Sort-merge join: R ⋈ S

- Scan R and sort in main memory
- Scan S and sort in main memory
- Merge R and S

```
do{
    if (!mark) {
        while (r < s) {advance r}
        while (r > s) {advance s}
        mark = s
}
If (r == s) {
    Result = <r, s>
    Advance s
    Return result
}
Else {
    Reset s to mark
    Advance r
    Mark = NULL
    }
}
```

$Cost =$ Sort R + Sort S + ([R]+[S])

c. (15) Describe a hash-join algorithm. Give the cost of joining $R$ and $S$ with a hash join.

A simple hash join is performed in two phases. In the building phase, the inner relation is hashed into main memory. The join attributes are used as a hash key. The second phase called probing phase is performed. In the probing phase, the outer relation is read sequentially and for each record in the outer relation the matching records in the inner relation are retrieved. Probing can be done at a constant cost because the inner relation is now in memory and has a hash access path on the join attributes.

$$Cost = 3(B_R + B_S)$$

$$Cost = 3(10000 + 2000)$$

$$\boldsymbol{Cost = 36000}$$

$$Seeks = 2\left(\frac{B_R}{B_B}\right) + \left(\frac{B_S}{B_B}\right)$$

$$Seeks = 2\left(\frac{10000}{52}\right) + \left(\frac{2000}{52}\right)$$

$$\boldsymbol{Seeks = 423.076}$$