WSU-CPTS-415

Charles Nguyen – 011606177

Homework 1

1. Volume, Velocity, Variety, Veracity, Value
   Example: Atmospheric Radiation Measurement (ARM) user facility at the Pacific Northwest National Lab (PNNL). I worked with a team of scientists there during my Capstone to investigate the possibility of writing API for querying on a data collection of 1000TB.
   - Volume: 1000TB
   - Velocity: Near real-time; a time series is so large that they must be computed on cloud server and only results are sent back to user.
   - Variety: Data is mostly structured, but some part was unstructured due to both old and new measurement devices.
   - Veracity: Data is trusted but the degrees of accuracy and precision are completely dependent on measurement devices.
   - Value: Highly meaningful meteoric information that helps shape socio-economic policies for the Pacific Northwest and the nation at large.

2.
   a. Terms:

- **Relation schema**: informally a data table; formally a set of attributes each with domain
    - `airports` is a *relation schema*, wherein its attributes are `AirportID`, `Name` and so on.
- **Relational database schema**: the layout that reflects the relations among multiple *relation schemas*; transcendent over *relation schema*
    - `airports`, `airline ` and `route ` compose a *relational database schema*.
- **Domain**: a range of all possible values, which itself must be defined with a specific data type and format with the possibility for NULL value.
    - `Latitude` and `Longitude` both belong to one domain and are both a double type ≥ 0 and cannot be NULL.
- **Attribute**: informally a column in the table; formally a particular feature in the set of attributes
    - `Name`, `City`, `Altitude` and so on are all *attributes* of the *relation schema* `airports`.
- **Attribute domain**: defines the role of a domain in a *relation schema*.
    - `Latitude` and `Longitude` are independent attributes with *attribute domains* orthogonal to each other.
- **Relation instance**: informally a set of rows of selected attributes in the table; formally an ordered n-tuple of values, each derived from a proper domain.
    - `FROM airports SELECT AirportID, Name, City, Country INTO location OFFSET 5;`

1	Goroka Airport	Goroka	Papua New Guinea
2	Madang Airport	Madang	Papua New Guinea
3	Mount Hagen Kagamuga Airport	Mount Hagen	Papua New Guinea
4	Nadzab Airport	Nadzab	Papua New Guinea
5	Port Moresby Jacksons International Airport	Port Moresby	Papua New Guinea

b. **Relational database schema** and **Relation schemas** for `airports`, `airline` and `route`:
   - Airport:
     - AirportID (PK)
     - Name
     - City
     - Country
     - IATA
     - ICAO
     - Latitude
     - Longitude
     - Altitude
     - Timezone
     - DST
     - Tz database time zone
     - Type
     - Source
   - Airline:
     - AirlineID (PK)
     - Name
     - Alias
     - IATA
     - ICAO
     - Callsign
     - Country
     - Active
   - Route:
     - Airline
     - AirlineID (FK=Airline.AirportID, PK1)
     - SourceAirport
     - SourceAirportID (FK=Airport.AirportID, PK2)
     - DestinationAirport
     - DestinationAirportID (FK=Airport.AirportID, PK3)
     - Codeshare
     - Stops
     - Equipment

Functional dependencies: all primary keys should have full functional dependencies over the remaining attributes in the table. In other words, given a foreign key (which is also a primary key pointing to another table), a query should be able to retrieve at most all information from the second table. For example, a query in Route using the foreign key AirlineID should be able to extract any and all information in the table Airline.

3. Functional Dependencies:
    a. Armstrong's:
        i. Reflexive: Given the primary key AirlineID and we select some columns into a new table, all instances in the new table should have the same values as all instances in the table Airline.

$$NewTable \subseteq Airline$$

        ii. Augmentation: In the table Route, using AirlineID to query only Name and Alias into a new table NameAlias. Using AirlineID again to query Callsign, Country and Active into a new table CCA. Every single attribute in either is fully dependent on AirlineID. Therefore, joining the two table makes a new table that is a proper subset of Airline.

$$(NameAlias + CCA) \subset Airline$$

    b. Proofs:
        i. Decomposition**: if $\boldsymbol{X \rightarrow YZ}$ then: $\boldsymbol{X \rightarrow Y}$ and $\boldsymbol{X \rightarrow Z}$**.
        From *a.ii.* above, we can see that the table $(NameAlias + CCA)$ can be decomposed back into $NameAlias$ and CCA by making appropriate queries from $(NameAlias + CCA)$.
            1. $X \rightarrow YZ$
            2. $YZ \rightarrow Y$ (Reflexive on $X \rightarrow YZ$)
            3. $X \rightarrow Y$ (Transitive on 1 and 2)

        ii. Pseudo transitivity: **if $\boldsymbol{X \rightarrow Y}$ and $\boldsymbol{YW \rightarrow Z}$ then: $\boldsymbol{XW \rightarrow Z}$**.
            1. $X \rightarrow Y$
            2. $WY \rightarrow Z$
            3. $WX \rightarrow WY$ (Augmenting W to $X \rightarrow Y$)
            4. $WX \rightarrow Z$ (Applying transitivity on 3 and 2)

4. Normalization

Given:

$R(A_1, A_2, A_3, A_4)$
$F_{one}: A_2, A_3 \rightarrow A_4$
$F_{two}: A_3, A_4 \rightarrow A_1$
$F_{three}: A_1, A_2 \rightarrow A_3$

3NF and BCNF:

- Using augmentation on F_{one}:
  - $(A_1 \cdot F_{one}) = \{A_1, A_2, A_3\} \rightarrow (A_1A_4 \rightarrow A_4 \; or \; A_1A_4 \rightarrow A_1)$ (decomposition)
  - $(A_1 \cdot F_{one}) \rightarrow A_4 \; where \; key$: (A_2, A_3) (3NF)
  - Given $(A_1A_2) \rightarrow A_3$, we have: $R_1(A_1, A_2, A_3)$ in BCNF
- Using augmentation on F_{two}:
  - $(F_{two} \cdot A_2) = \{A_3, A_4, A_2\} \rightarrow (A_1A_2 \rightarrow A_1 \; or \; A_1A_2 \rightarrow A_2)$ (decomposition)
  - $(F_{two} \cdot A_2) \rightarrow A_1 \; where \; key$: (A_3, A_4) (3NF)
  - Given $(A_2, A_3) \rightarrow A_4$, we have: $R_2(A_2, A_3, A_4)$ in BCNF
- Using augmentation on F_{three}:
  - $(F_{three} \cdot A_4) = \{A_1, A_2, A_4\} \rightarrow (A_3A_4 \rightarrow A_1)$
  - Already in F_{two} so we discard.