# NATIONAL INSTITUTE OF TECHNOLOGY KARNATAKA

## Department of Information Technology

Advanced Database Systems

Assignment 1

**Distributed Geographic Database System for forest related activities.**

Name : Pritish Uday Naik

Roll No : 192422IT010

Class : M.Tech 1st year

# Contents

# 1. Problem Description:

Information plays a vital role in the development and growth of mankind. Currently, the various departments manage information related to forest independently in their own ways .There are no common, standardized process and program for capturing, processing and storing forest information. This has kept forest information disintegrated in different departments. The various departments have systems in place to store and process data related to forest but the systems are not able to talk to each other. This makes it difficult for the Wildlife NGOs, Businessmen and Forest conservation officer to collate information related to forest across departments. For instance,consider the state is divided into many state divisions. Forest is divided into forest type which contains many state divisions and each state division has unique division_id. If the Forest conservation officer wants information to know the reason for the sudden reduction in the number of a particular tree species urgently, he may go to the water resource department to collect information about water quality and quantity that is used in his state_division. He may also go to Police department to collect the data related to smuggling and poaching cases in his state_division. On occasions where the department is not able to produce the needed information immediately, the business or activity at that particular time would come to a standstill. On the other hand, time is being wasted going round the various departments to solicit data. This situation is very frustrating and impedes smooth operations and decision making process. Similar is the case with NGO activists and Businessmen.

**Requirement of Wildlife NGO representative:**
Wildlife NGO representative said "My organisation's objective is to educate the society and foster scientific research in the field of forest. As a representative of NGO, I want you to build a system on the following guidelines. On the first page I want list of all forest types (i.e. Evergreen forest, Deciduous forest, Montane grassland) in the state. Then I would choose the name of the forest that I am interested in from a drop-down list. After pressing the button 'Search', I should get a list of the available options for state_divisions and forest officer id and Forest officer name. After I choose one option from the list of state_divisions, I should get a list of tree species along with the count(number of tree species) in the last survey made by forest patrollers. I should be able to check more details related to wildlife surveys done by the forest department. For that I want to click the button ‘check wildlife surveys’. There I would get a details of the total number of camps, total number of patrollers. I arrange tree plantation activities in the forest state_division.So, I need details about the water resources in the forest state_division. I should have ‘Check water resources here’ button to get the details like water resource types (underground water, ponds, lakes, canals, river, stream), available water capacity at each resource and contamination level like carcinogenicity and toxicity."

**Requirement of businessmen:**
There are many industries like saw-mill industry, broom industry and furniture industry, perfume industry, etc which depend upon forest. Every businessman who wants raw material from forest should choose forest type (i.e. Evergreen forest, Deciduous forest, Montane grassland)in the state. After pressing the button 'Search' he will get a list of the available options for Forest state_divisions. After that he has to choose one option from each of these categories, he gets a list of tree species along with count of tree species. The raw material will be collected by the government and kept for auction at depots. Below the screen there is additional information. Here he will find the contact person for the auction, auction_location. If any businessman wants to procure the raw material from government depots through public auction he should be able to

know the depot location. The businessman may be interested about illegal activities in the state_division. There will be a button at the top left corner called 'Check illegal activities'. Here, he will find the cases related to illegal activities related to forest in the state division. The businessman who has the furniture industry will be interested in bamboo and may want to know whether there are dry bamboo trees in the region. If there are dry bamboos, there is a high possibility of forest fire. In case there is a fire, are there any fire stations in the state_divisions and do they have water stock.The perfume industry businessman who wants to buy sandalwood should be able to check whether there are any sandalwood smuggling cases in the region. The broom industry businessman who is interested in raw materials related to grasslands wants to check the number of illegal grazing activities and fire cases in the region of his interest.

**Requirement of Forest conservation officers:**

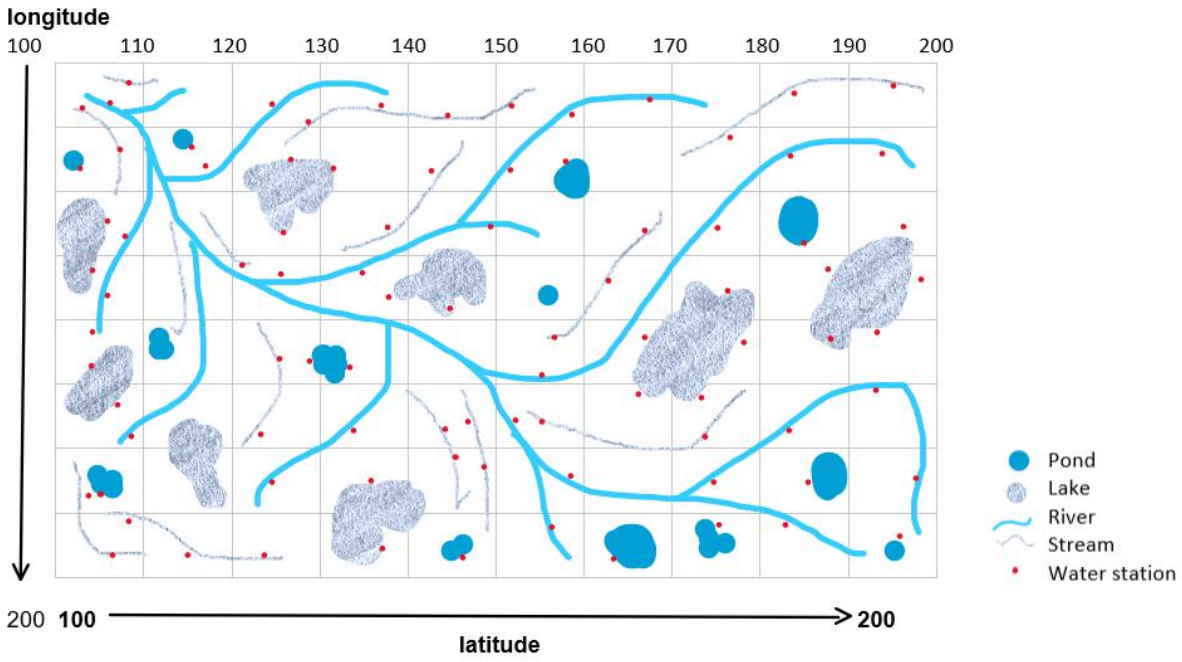
Forest officers conserve and sustainably manage forest resources with involvement of village communities living in the vicinity of forests. Protection of forest land from permanent loss is the most important duty of any forest officer.

Forest officers are responsible for the collection and credit of Forest Revenue, investigate Forest Offences in their division. Every state_division will have one forest officer. Forest officer accesses the wildlife survey data like total number of camps, total number of patrollers,count of tree species, and species status. If the number of patrollers in the state_division is not sufficient, the forest officer can increase wildlife patrollers for wildlife surveys. There could be many illegal activities happening in his state_division like encroachment, illegal grazing in grasslands , wild animals poaching and smuggling. Forest officer is responsible for keeping account of the Forest encroachments and illegal grazing activities and deal with all such cases. The smugglers and poaching offenders will be arrested by the armed state police which operates in division. Forest officers periodically checks the State police data to get the details of the crime like smuggling and poaching related to the forest.

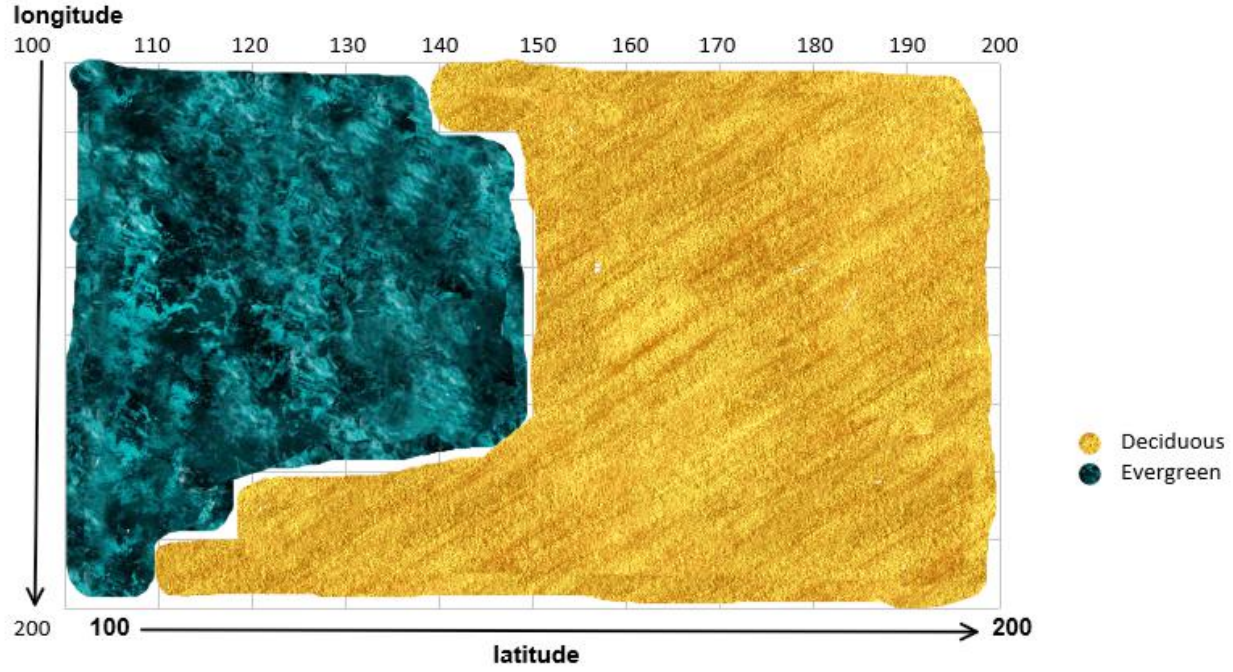
From the wildlife surveys, the forest officer can deduce the possibility of fire. So Forest officer checks all fire station names, fire station location, and fire station administrator and water stock at each fire station. The forest officer checks the available quantity of water at water resource and whether that is sufficient in the given scenario. If not sufficient, he can request the water department to increase quantity in the water resource.

# 2. Data sources:

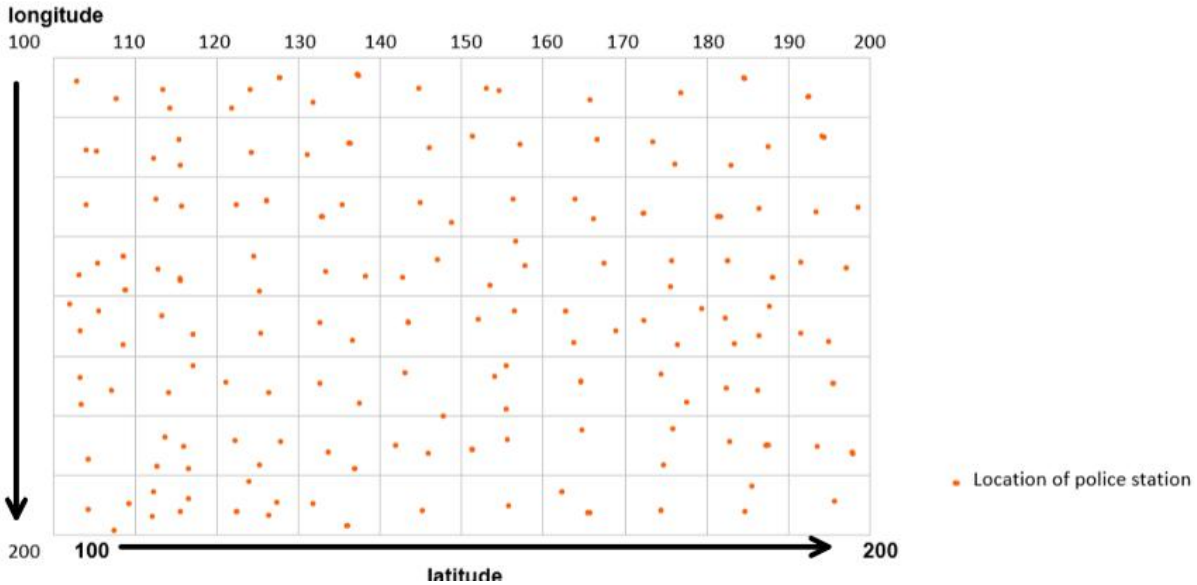
**Water resource department:**
The objectives of this department is to analyze the different water resources and recharge zones of water resources and monitor the contamination of water resources.The state has 100 divisions. Each division in the state has at least one water resource. Each water resource in the division will have one dedicated station. Each station will have a unique station_id, station_name, station_location(longitude, latitude), water resource name, quantity of stored water at station, division_id. Each station periodically takes sample of water resource.Each sample in a station will have unique sample id. The analysed sample will have station id,sample_id, Date_of_analysis, time_of_analysis, method_of _analysis, concentration, analysis_method, toxicity, carcinogenicity.



**Forest department:**
The forest department has an objective to conserve and expand unique and complex natural forests in the state without affecting their ecological processes, increase the tree cover both inside and outside the forest and meet the forest product demands of the society.The state has 100 divisions.The state has 2 types of forest i.e Evergreen forest, Deciduous forest. The state has 2 different forest names viz Kudremukh National Park, Bandipur National Park. Each division will have only one type of forest. Each division in the state will have state division identification division_id, forest_name, forest_officer_id, forest_station_id, Forest_type, Forest_area, revenue generated in his state division, Auction location of the state division, contact number for auction. Each Forest state division will have one auction depot, one forest officer and one forest station. Every division in the forest periodically performs wildlife survey. The wildlife survey is performed by forest patrollers. Wildlife survey has division_id, number_of_patrollers, number_of_survey _camps, count of each important tree species (bamboo, sandalwood,teakwood,timber,grass).There will be only one forest station in a division. Each Forest station has forest dispute registry which has case id, case type ( encroachment, illegal grazing etc), accused_name, area of land affected, forest station id.



**Police department:**
The Police have statutory powers to investigate crimes, apprehend offenders, interrogate and prosecute suspects. They are also empowered to search and seize properties suspected to be stolen or associated with smuggling of forest products like sandalwood, teakwood,rosewood, other valuable timber species and poaching of animals like elephants for tusks.Every division in state will have at least one police station. Each police station in a division will have unique police_station_id, police_station_name, location(latitude, longitude), division_id, police_officer_id. Each Police station maintains a crime registry containing Crime_id, crime type, Date_of_offence, Date_of_arrest, FIR_no, under_section, Accused_name, police_station_id . Crime_id in a crime registry is unique for a given police station.




**Fire department:**
The objective of Fire services is to protect the society and nature from fire by providing progressive high quality emergency services and preventive measures.The fire department database will have many fire stations.Each state division will have at least one fire station. Each fire station will station_id, station_name, state division_id in which fire station belongs, name of the administrative head of fire station, location (longitude,latitude), number of forest fire cases in the division, Water stock available at the station, name of water resource from which water is fetched. Each fire station gets water from only one water resource

# 3. Queries

Wildlife NGOs Queries:

1. List the Division id for a given Forest type and forest name
2. List the divisions for a given forest type and forest name that has the highest number of a given species
3. The total quantity of Water at water resource in a forest for a given forest type and forest name

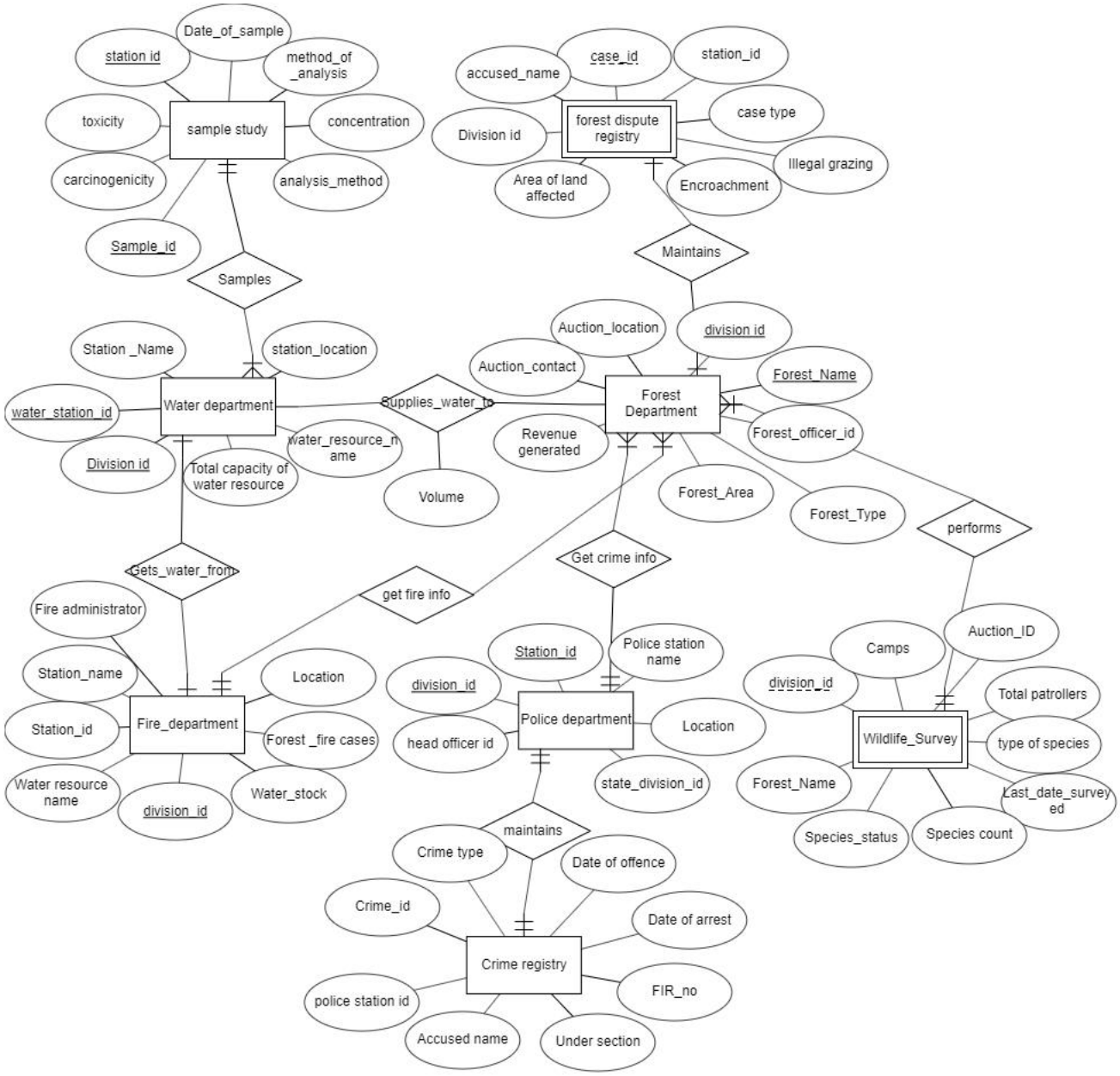
Businessmen Queries:

4. For a given forest type and forest name get the contact number and location of auction depot in a division which has the highest number of a given species.
5. For a given Forest type and Forest name, Find the number of cases registered at a police station in the division for a particular crime type in ascending order.

Forest officer Queries:

6. For a given Forest type and Forest name and a division(s) which has the count of a bamboo species above threshold value, find total water stock available at all fire stations in that division
7. For a given Forest type and Forest name and a division which has count of a bamboo species greater than a threshold value , List the toxicity of water at a water resource used by the Fire station.
8. For a given Forest type and Forest name and divisions which have count of a bamboo species greater than threshold value , list the total quantity of water at a given water resources used by fire stations in that forest.

# 4. EER Model (Conceptual Model)



# 5. Global Conceptual Schema:

The transformation from the entity-relationship model to the relational model is very straightforward. A feasible set of relational schemas is as follows.

**Water Department:**

| Water_station_id | Water_station_name | Water_station_location | water_rsc_name |
|---|---|---|---|
| stored water | div_id | | |

**Water Quality:**

| Water_Station_id | Water_sample_id | Date_of_analysis | time_of_analysis | method_of_analysis |
|---|---|---|---|---|
| Concentration | Toxicity | carcinogenicity | | |

**Police department:**

| police_station_id | police_station_name | location(latitude, longitude) | div_id | police_officer_id. |
|---|---|---|---|---|

**Police crime registry:**

| Crime_id | Crime_type | Date_of_offence | Date_of_arrest | FIR_no | under_section |
|---|---|---|---|---|---|
| Accused_name | Crime_id | | | | |

**Forest Department:**

| div_id | forest_name, | forest_offr_id, | forest_stn_id | Forest_type | Forest_area | revenue |
|---|---|---|---|---|---|---|
| Auction_location | Auction_contact | | | | | |

**Wildlife survey:**

| div_id | no_of_patrollers | no_of_survey_camps | bamboo count | sandalwood count |
|---|---|---|---|---|
| teakwood count | timber count | grass count | Species status | |

**Forest dispute registry:**

| Dispute_case id | Dispute_case_type | accused_name | Area_of_land_affected | Forest_station id |
|---|---|---|---|---|

**Fire department**

| Fire_Station_id | Fire_station_name | div_id | Fire_station_head | Fire_station_Location |
|---|---|---|---|---|
| Forest_fire_cases | Water stock | Fetched_from | | |

# 6. Instance of each relation:

**Water Department:**

| W_stn_id | div_id | W_stn_name | W_stn_loc_long | W_stn_loc_lat | W_rsc_name | Stored_water (lit.) |
|---|---|---|---|---|---|---|
| 01 | 01 | yamuna | 100 | 101 | stream | 25000 |
| 02 | 01 | bramha | 100 | 102 | stream | 10000 |
| 03 | 01 | chatur | 100 | 102 | river | 100030 |
| 01 | 02 | ramalinga | 100 | 111 | pond | 234290 |
| 02 | 02 | Anakala | 101 | 112 | stream | 87695 |
| 03 | 02 | kantha | 103 | 114 | river | 2341254 |
| 01 | 03 | vaidehi | 106 | 122 | pond | 235426 |
| 02 | 03 | Quetta | 104 | 121 | river | 26542 |
| 03 | 03 | jamula | 101 | 129 | lake | 238423 |

**Water Quality check:**

| W_stn_id | div_id | W_sample_id | date_of_analysis | time_of_analysis | method_of_analysis | concentration(%) | toxicity(%) | carcinogenicity(%*) |
|---|---|---|---|---|---|---|---|---|
| 01 | 01 | 01 | 25-09-2019 | 12:30 | Chemical | 20 | 10 | 9 |
| 01 | 01 | 02 | 26-09-2019 | 1:45 | Electrical | 21 | 14 | 5 |
| 02 | 01 | 01 | 21-09-2019 | 12:30 | Electrical | 30 | 11 | 7 |
| 02 | 01 | 02 | 25-09-2019 | 2:30 | Electrical | 20 | 10 | 9 |
| 03 | 01 | 01 | 21-09-2019 | 12:10 | Electrical | 30 | 11 | 7 |
| 03 | 01 | 02 | 25-09-2019 | 9:30 | Chemical | 20 | 10 | 9 |
| 01 | 02 | 01 | 11-09-2019 | 2:10 | Electrical | 10 | 11 | 7 |
| 01 | 02 | 02 | 15-09-2019 | 10:30 | Chemical | 22 | 11 | 9 |
| 02 | 02 | 01 | 21-09-2019 | 12:30 | Chemical | 30 | 11 | 7 |
| 02 | 02 | 02 | 25-09-2019 | 2:30 | Electrical | 20 | 10 | 9 |
| 03 | 02 | 01 | 21-09-2019 | 12:10 | Chemical | 30 | 11 | 7 |
| 03 | 02 | 02 | 25-09-2019 | 9:30 | Chemical | 20 | 10 | 9 |
| 01 | 03 | 01 | 11-09-2019 | 2:10 | Electrical | 10 | 11 | 7 |
| 01 | 03 | 02 | 15-09-2019 | 10:30 | Chemical | 22 | 11 | 9 |
| 02 | 03 | 01 | 21-09-2019 | 12:30 | Chemical | 30 | 11 | 7 |
| 02 | 03 | 02 | 25-09-2019 | 2:30 | Electrical | 20 | 10 | 9 |
| 03 | 03 | 01 | 21-09-2019 | 12:10 | Chemical | 30 | 11 | 7 |
| 03 | 03 | 02 | 25-09-2019 | 9:30 | Electrical | 20 | 10 | 9 |

**Police department:**

| P_stn_id | div_id | P_stn_name | P_stn_loc_long | P_stn_loc_lat | P_officer_id. |
|---|---|---|---|---|---|
| 01 | 01 | lalbag | 101 | 102 | 01 |
| 02 | 01 | parel | 102 | 105 | 02 |
| 01 | 02 | santacruz | 104 | 110 | 01 |
| 02 | 02 | washi | 106 | 113 | 02 |
| 03 | 02 | talegaon | 106 | 117 | 03 |
| 01 | 03 | kashi | 107 | 121 | 01 |
| 02 | 03 | kase | 105 | 124 | 02 |

**Police criminal registry:**

| crime_id | P_station_id | div_id | crime_type | Date_of_offence | Date_of_arrest | FIR_no | under_section | Accused_name |
|---|---|---|---|---|---|---|---|---|
| 01 | 01 | 01 | Smuggling | 15-09-2019 | 17-10-2019 | 100 | IPC213 | Suresh |
| 02 | 01 | 01 | Theft | 16-09-2019 | 21-11-2019 | 100 | IPC213 | Ramesh |
| 03 | 01 | 01 | Smuggling | 21-09-2019 | 27-09-2019 | 100 | IPC213 | Rajesh |
| 01 | 02 | 01 | Smuggling | 25-09-2019 | 27-09-2019 | 100 | IPC213 | Prathamesh |
| 01 | 03 | 01 | Smuggling | 21-09-2019 | 17-10-2019 | 100 | IPC213 | Mahesh |
| 02 | 03 | 01 | Smuggling | 19-09-2019 | 27-09-2019 | 100 | IPC213 | Yogesh |
| 01 | 01 | 02 | Smuggling | 05-09-2019 | 27-09-2019 | 100 | IPC213 | Sagar |
| 02 | 01 | 02 | Theft | 29-09-2019 | 30-09-2019 | 100 | IPC213 | Samir |
| 03 | 01 | 02 | Theft | 25-10-2019 | 27-09-2019 | 100 | IPC213 | Amir |
| 01 | 02 | 02 | Smuggling | 25-09-2019 | 27-09-2019 | 100 | IPC213 | Sundar |
| 01 | 03 | 02 | Theft | 25-09-2019 | 27-09-2019 | 100 | IPC213 | Bandar |
| 02 | 03 | 02 | Smuggling | 25-09-2019 | 27-09-2019 | 100 | IPC213 | Gangadhar |

**Forest Department:**

| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 01 | Bandipur | 01 | 01 | Evergreen | 2103421 | 109 | 105 | 8732183452 |
| 02 | Bandipur | 02 | 02 | Deciduous | 2022143 | 103 | 115 | 7826183442 |
| 03 | Kudremukh | 03 | 03 | Evergreen | 7392621 | 108 | 121 | 5628123983 |
| 04 | Kudremukh | 04 | 04 | Deciduous | 2918361 | 106 | 132 | 8492018235 |

**Wildlife survey:**

| div_id | no_of_patrollers | no_of_survey_camps | bamboo_count | sandalwood count | teakwood count | timber count | grass count |
|---|---|---|---|---|---|---|---|
| 01 | 105 | 10 | 104 | 2984 | 198432 | 124235 | 198729 |
| 02 | 43 | 29 | 6765 | 8463 | 1292 | 12863 | 192045 |
| 03 | 65 | 21 | 192820 | 1629 | 1207 | 123 | 871 |

**Forest dispute registry:**

| case_id | F_stn_id | case_type | accused_name | area of land affected(km Sq.) |
|---|---|---|---|---|
| 01 | 01 | encroachment | Ajay | 10 |
| 02 | 01 | illegal grazing | Sanjay | 2 |
| 03 | 01 | encroachment | Jay | 13 |
| 01 | 02 | illegal grazing | Samar | 12 |
| 01 | 03 | encroachment | Wahir | 4 |
| 02 | 03 | illegal grazing | Tahir | 20 |
| 03 | 03 | encroachment | Jay | 3 |

**Fire department**

| Fire_stn_id | div_id | Fire_stn_head_name | Fire_stn_loc_lat | Fire_stn_loc_long | Forest_fire_cases | Water_stock(lit.) | Fetched_from(W_stn_id) |
|---|---|---|---|---|---|---|---|
| 01 | 01 | Samir | 105 | 100 | 121 | 2983 | 02 |
| 02 | 01 | Qayamat | 109 | 101 | 19 | 2341 | 01 |
| 01 | 02 | Amir | 104 | 110 | 21 | 1321 | 01 |
| 01 | 02 | Shahrukh | 105 | 116 | 421 | 1521 | 02 |
| 01 | 03 | Akshay | 102 | 120 | 231 | 1351 | 01 |

# 7. Normalisation:

Goals of normalization

1. Integrity
2. Maintainability

Benefits (Side effects) of normalization

* Reduced storage space required (usually, but it could increase)
* Simpler queries (sometimes, but some could be more complex)
* Simpler updates (sometimes, but some could be more complex)

First normal form (1NF) -- a table R is in 1NF iff all underlying domains contain only atomic values, i.e. there are no repeating groups in a row.
Following are the 1NF of all the relations

**Water Department:**

| Water_station_id | Water_station_name | Water_station_location_latitude | |
|---|---|---|---|
| Water_station_location_longitude | water_resource_name | stored_water | division_id |

**Water Quality:**

| Water_Station_id | division_id | Water_sample_id | Date_of_analysis | time_of_analysis |
|---|---|---|---|---|
| method_of_analysis | Concentration | Toxicity | carcinogenicity | |

**Police department:**

| police_station_id | division_id | Police_station_loc_lat | Police_station_location_long |
|---|---|---|---|
| police_station_name | police_officer_id. | | |

**Police crime registry:**

| Crime_id | Police_station_id | Division_id | Crime_type | Date_of_offence | Date_of_arrest |
|---|---|---|---|---|---|
| FIR_no | under_section | Accused_name | Crime_id | | |

**Forest Department:**

| division_id | forest_name | forest_offr_id | forest_stn_id | Forest_type | Forest_area | revenue | |
|---|---|---|---|---|---|---|---|
| Auction_location | Auction_contact | | | | | | |

**Wildlife survey:**

| division_id | no_of_patrollers | no_of_survey_camps | bamboo count | sandalwood count |
|---|---|---|---|---|
| teakwood count | timber count | grass count | | |

**Forest dispute registry:**

| Dispute_case id | Dispute_case_type | accused_name | Area_of_land_affected | Forest_station id |
|---|---|---|---|---|

**Fire department**

| Fire_Station_id | div_id | Fire_station_name | Fire_station_head | Fire_station_Loc_latitude |
|---|---|---|---|---|
| Fire_station_Loc_longitude | Forest_fire_cases | Water stock | Fetched_from | |

Functional dependency given a table R, a set of attributes B is functionally dependent on another set of attributes A if at each instant of time each A value is associated with only one B value.
This is denoted by A -> B. A trivial FD is of the form XY --> X (subset).

super-key -- a set of one or more attributes, which, when taken collectively, allows us to identify uniquely an entity or table.
candidate key—any subset of the attributes of a super-key that is also a super-key, but not reducible.
primary key -- arbitrarily selected from the set of candidate keys, as needed for indexing.

Functional dependencies on relations:
The underlined attributes are primary keys:

**Water Relation:**
Water_station_id,division_id ----> Water_station_name, Water_station_location_latitude, Water_station_location_longitude water_resource_name,stored_water

**Water station relation:**
Water_Station_id, division_id, Water_sample_id, ------> Date_of_analysis, time_of_analysis, method_of_analysis,Concentration, Toxicity, Carcinogenicity

**Police relation:**
police_station_id, division_id ------> Police_station_loc_lat, Police_station_location_long police_station_name, police_officer_id.

**Police Crime relation:**
Crime_id, Police_station_id, Division_id -------> Crime_type, Date_of_offence, Date_of_arrest, FIR_no, Under_section, Accused_name, Crime_id

**Forest Relation:**
division_id, ------> forest_name, forest_offr_id, forest_stn_id, Forest_type, Forest_area, revenue, Auction_location  Auction_contact

**Wildlife survey relation:**
division_id ------> no_of_patrollers, no_of_survey_camps, bamboo count, sandalwood count, teakwood count, timber count , grass count

**Forest dispute relation:**
Dispute_case id, Forest_station id -------> Dispute_case_type, accused_name, Area_of_land_affected

**Fire Station relation:**
Fire_Station_id, division_id, --------> Fire_station_name, Fire_station_head, Fire_station_Loc_latitude, Fire_station_Loc_longitude, Forest_fire_cases, Water stock Fetched_from

Now we can apply 3rd Normal form formula

Third normal form (3NF)
A table is in 3NF if, for every nontrivial FD X --> A, either:
(1) attribute X is a super-key, or
(2) attribute A is a member of a candidate key (prime attribute)

From the functional dependencies it can be observed that all the attributes of all the relations are functionally dependent only on Primary key and nothing else, all the relations are already in 3rd Normal Form.

# 8. Global Schema:

| Water Department | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Water_station_id | int(2) |
| Water_station_name | char(10) |
| Water_station_location_latitude | int(2) |
| Water_station_location_longitude | int(2) |
| water_resource_name | char(10) |
| stored_water | char(10) |
| division_id | int(2) |

| Water Quality | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Water_station_id | int(2) |
| Water_sample_id | int(2) |
| Date_of_analysis | int(2) |
| Time_of_analysis | int(2) |
| Method_of _analysis | char(10) |
| Toxicity | int(2) |
| division_id | int(2) |
| Carcinogenicity | int(2) |

| Police department | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Police_station_id | int(2) |
| Police_station_name | char(10) |
| Police_station_location_latitude | int(2) |
| Police_station_location_longitude | int(2) |
| division_id | int(2) |
| police_officer_id. | char(10) |

| Police crime registry | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Crime_id | int(2) |
| Crime_type | char(10) |
| Date_of_offence | int(2) |
| Date_of_arrest | int(2) |
| FIR_no | int(2) |
| under_section | char(10) |
| Accused_name | char(10) |
| police_station_id | int(2) |

| Forest department | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| div_id | int(2) |
| Forest_name | char(10) |
| Forest_offr_id | int(2) |
| Forest_stn_id | int(2) |
| Forest_type | int(2) |
| Forest_area | char(10) |
| Revenue | char(10) |
| Auction_location_latitude | int(2) |
| Auction_location_longitude | int(2) |
| Auction_contact | int(2) |

| Wildlife surveys | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| div_id | int(2) |
| no_of_patrollers | int(2) |
| no_of_survey_camps | int(2) |
| bamboo count | int(2) |
| sandalwood count | int(2) |
| teakwood count | int(2) |
| timber count | int(2) |
| grass count | int(2) |

| Forest Dispute registry | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Dispute_case_id | int(2) |
| Dispute_case_type | int(2) |
| accused_name | char(10) |
| area_of_land_affected | int(2) |
| forest_station_id | int(2) |

| Fire Department | |
|---|---|
| **Attribute name** | **Attribute size (type in bytes)** |
| Station_id | int(2) |
| station_name | char(10) |
| div_id | int(2) |
| station_head_name | char(10) |
| Fire_station_Location_latitude | int(2) |
| Fire_station_Location_longitude | int(2) |
| Forest_fire_cases | int(2) |
| Water_stock | int(2) |
| Fetched_from | int(2) |

# 9. Fragmentation:

Data fragmentation is division of relation R into fragments R1,R2,R3,....Rn which contains sufficient information to reconstruct R

There are 3 methods for fragmentation:

1. Horizontal fragmentation :
   Tuples of R is assigned to a fragment
2. Vertical fragmentation
   Schemas of R is split into sub-schemas such that all sub-schema should contain a candidate key to ensure lossless join property. A special attribute, tuple id may be added to each subschema to serve as a candidate key

Correctness of fragmentation: Fragmentation cannot be carried out haphazardly. There are three rules that must be followed during fragmentation:

1. Completeness:
   If a relation instance R is decomposed into fragments R1, R2, . . . , Rn,each data Item that can be found in R must appear in at least one fragment. This rule is necessary to ensure that there is no loss of data during fragmentation.
2. Reconstruction:
   It must be possible to define a relational operation that will reconstruct the relation R from the fragments. This rule ensures that functional dependencies are preserved.
3. Disjointness:
   If a data item di appears in fragment Ri, then it should not appear in any other fragment. Vertical fragmentation is the exception to this rule, where primary key attributes must be repeated to allow reconstruction. This rule ensures minimal data redundancy.

In the case of horizontal fragmentation, a data item is a tuple; for vertical fragmentation, a data item is an attribute.

Advantages of fragmentation

Horizontal:

1. Allows parallel processing on fragments of a relation
2. Allows relation to be split so that tuples are located where they are most frequently used

Vertical:

1. Allows tuples to be split and stored where it is most frequently used
2. Tid is allows efficient joining of vertical fragments

Disadvantages

1. The performance of global applications that require data from several fragments located at different sites may be slower.
2. Integrity control may be more difficult if data and functional dependencies are fragmented and located at different sites.

Following are the detailed SQL queries in the geographic forest management system

1. **List the Division id for a given Forest type and forest name**

```
SELECT div_id FROM Forest Department WHERE Forest.forest_type = "Evergreen" ,
Forest.forest_name    = “Bandipur National Park”
```

2. **List the divisions for a given forest type and forest name that has highest number of a given species**

```
SELECT Forest.div_id,  max(Wildlife.sandlewood_count)
FROM Forest
JOIN Wildlife
ON Forest.div_id=Wildlife.div_id
WHERE Forest.forest_type = "Evergreen" , Forest.forest_name = “Bandipur National
Park”
```

3. **The total quantity of Water at water resource in a forest for a given forest type and forest name**

```
SELECT  sum(Water.Stored_water)
FROM Forest
JOIN Water
ON Forest.div_id=Water.div_id
WHERE Forest.forest_type = "Evergreen" and Forest.forest_name = “Bandipur National
park” and Water.Water_resource_name = 'lake'
```

4. **For a given forest type and forest name get the contact number and location of auction depot in a division which has highest number of a given species.**

```
SELECT  Forest.F_auc_depo_loc_lat, Forest.F_auc_depo_loc_long,F_auc_contact ,
MAX(Wildlife.sandalwood_count)
FROM Forest
JOIN Wildlife
ON Forest.div_id=Wildlife.div_id
WHERE F_type = "Deciduous" and forest name = “Bandipur National Park”
```

5. **For a given Forest type and Forest name, Find the police station which has minimum number of smuggling cases registered**

```
SELECT  Police_station_id, div_id, MAX(crime_type)
FROM Forest
JOIN PoliceCrimeRegistry
ON Forest.div_id=PoliceCrimeRegistry.div_id
```

```
WHERE Forest.forest_type = "Evergreen" AND Forest.forest name = “Bandipur National
park” AND PoliceCrimeRegistry.crime_type = "smuggling"
GROUPBY PoliceCrimeRegistry.Police_station_id,div_id
```

6. **For a given Forest type and Forest name and a division which has count of a bamboo species > 500 , find water stock available at a fire stations in that division**

```
SELECT  Fire.div_id, sum(Fire.Water_stock)
FROM Fire
JOIN Forest
ON Forest.div_id=Fire.div_id
JOIN Wildlife
ON Fire.div_id=Wildlife.div_id
WHERE Forest.forest_type = "Evergreen" and Forest.forest_name = “Bandipur National
park” AND WILDLIFE.Bamboo_count > 500
GROUPBY Fire.div_id
```

7. **For a given Forest type and Forest name and a division which has count of a bamboo species > 500 , List the Fire station which uses water from a water resource that has least toxicity**

```
SELECT  WaterQuality.MIN(toxicity)
FROM WaterQuality
JOIN Fire
ON WaterQuality.stn_id = Fire.Fetched_from  AND WaterQuality.div_id=Fire.div_id
JOIN Forest
ON WaterQuality.div_id=Forest.div_id
JOIN Wildlife
ON WaterQuality.div_id=Wildlife.div_id
WHERE Forest.forest_type = "Evergreen" and Forest.forest_name = “Bandipur National
forest” AND Wildlife.Bamboo_count > 500
GROUPBY Fire.stn_id
```

8. **For a given Forest type and Forest name and a division which has count of a bamboo species > 500 , list the water resources used by fire stations in that forest**

```
SELECT  Fire.stn_name, Water.water_rsr_name
FROM Water
JOIN Fire
ON Water.stn_id = Fire.Fetched_from  AND Water.div_id=Fire.div_id
JOIN Forest
ON Water.div_id=Forest.div_id
JOIN Wildlife
ON Water.div_id=Wildlife.div_id
WHERE Forest.forest_type = "Evergreen" and Forest.forest_name = “Bandipur National
park” AND Wildlife.Bamboo_count > 500 AND Fire.fire_station_id = "01"
```

**9.1 Horizontal Fragmentation:**

Horizontal fragmentation partitions the relation along its tuples of the relations. Every fragment will have the same number of attributes. There are two ways doing it. Primary and derived horizontal fragmentation.

**Primary Horizontal fragmentation on Forest Relation:**

From the Queries shown above we can deduce following minterm Predicates for Forest relation:

P1 : (Forest_name = “Bandipur National Park”) AND (Forest_type = ”Deciduous)
P2 : (Forest_name = “Bandipur National Park”) AND (Forest_type = ”Evergreen)
P3 : (Forest_name = “Kudremukh National Park”) AND (Forest_type = “Deciduous”)
P4 : (Forest_name = “Kudremukh National Park”) AND (Forest_type = ”Evergreen” )

Instance of Forest relation:

| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 01 | Bandipur | 01 | 01 | Evergreen | 2103421 | 109 | 105 | 8732183452 |
| 02 | Bandipur | 02 | 02 | Deciduous | 2022143 | 103 | 115 | 7826183442 |
| 03 | Kudremukh | 03 | 03 | Evergreen | 7392621 | 108 | 121 | 5628123983 |
| 04 | Kudremukh | 04 | 04 | Deciduous | 2918361 | 106 | 132 | 8492018235 |

Based on these predicates 4 fragments of Forest relation will be created:

Fragment 1: All the tuples of Forest relation that satisfy predicate P1

| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 01 | Bandipur | 01 | 01 | Evergreen | 2103421 | 109 | 105 | 8732183452 |

Fragment 2: All the tuples of Forest relation that satisfy predicate P2

| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 02 | Bandipur | 02 | 02 | Deciduous | 2022143 | 103 | 115 | 7826183442 |

Fragment 3: All the tuples of Forest relation that satisfy predicate P3

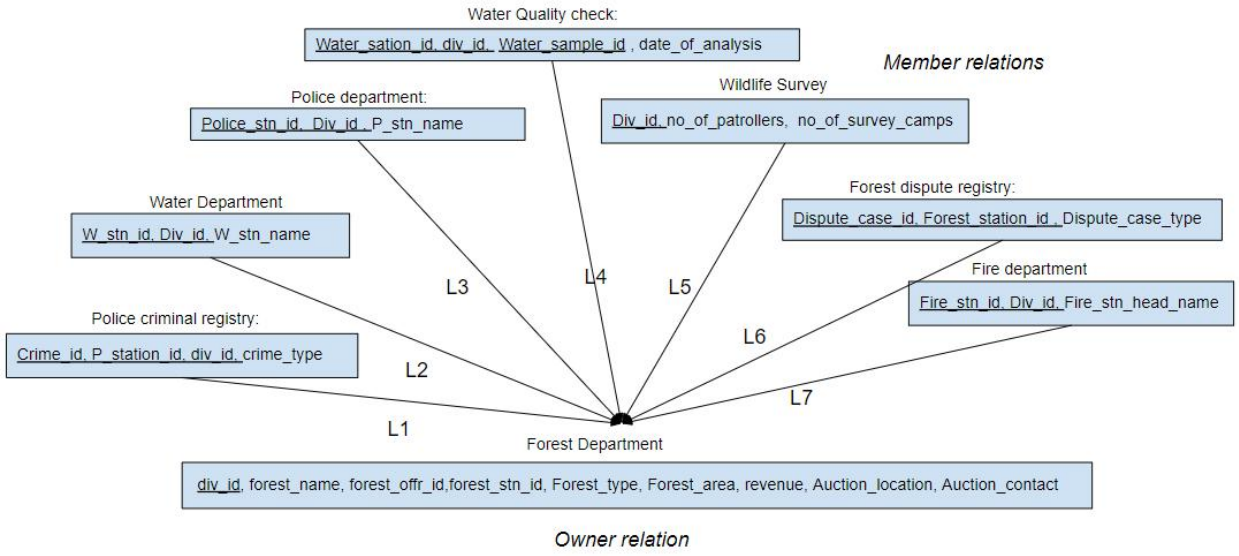
| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 04 | Kudremukh | 04 | 04 | Deciduous | 2918361 | 106 | 132 | 8492018235 |

Fragment 4: All the tuples of Forest relation that satisfy predicate P4

| div_id | F_name | F_offr_id | F_stn_id | F_type | F_revenue(rs) | F_auc_loc_lat | F_auc_loc_long | F_auc_contact |
|---|---|---|---|---|---|---|---|---|
| 04 | Kudremukh | 04 | 04 | Deciduous | 2918361 | 106 | 132 | 8492018235 |

**Derived horizontal fragmentation:**

Fragmentation definition of relation S derived from existing horizontal fragmentation of relation R using foreign key relationships.




Derived horizontal fragmentation is defined on a member relation of a link according to a selection operation specified on its owner. The important point to note is that each link (L1,L2,L3,L4,L5,L6,L7) is an equi-join. The equi-join can be implemented using semi-joins. Semi-join reduces the amount of data that needs to be transmitted between sites.

Owner : Forest relation

Member : Water relation, Water Quality relation, Police relation, Police Crime relation, Wildlife survey relation, Forest dispute registry relation, Fire relation.

Derived horizontal Fragments of Water department is:

Fragment 5 (Water relation) = Water relation ⋈ Fragment 1 (Forest relation)

Fragment 6 (Water relation) = Water relation ⋈ Fragment 2 (Forest relation)

Fragment 7 (Water relation) = Water relation ⋈ Fragment 3 (Forest relation)

Fragment 8 (Water relation) = Water relation ⋈ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Water Quality relation:

Fragment 9 (Water Quality relation) = Water Quality ⋈ Fragment 1 (Forest relation)

Fragment 10 (Water Quality relation) = Water Quality ⋈ Fragment 2 (Forest relation)

Fragment 11 (Water Quality relation) = Water Quality ⋈ Fragment 3 (Forest relation)

Fragment 12 (Water Quality relation) = Water Quality ⋈ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Police Department relation:
Fragment 13 (Police relation) = Police ⟕ Fragment 1 (Forest relation)
Fragment 14 (Police relation) = Police ⟕ Fragment 2 (Forest relation)
Fragment 15 (Police relation) = Police ⟕ Fragment 3 (Forest relation)
Fragment 16 (Police relation) = Police ⟕ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Police Crime registry relation:
Fragment 17 (Police Crime relation) = Police Crime ⟕ Fragment 1 (Forest relation)
Fragment 18 (Police Crime relation) = Police Crime ⟕ Fragment 2 (Forest relation)
Fragment 19 (Police Crime relation) = Police Crime ⟕ Fragment 3 (Forest relation)
Fragment 20 (Police Crime relation) = Police Crime ⟕ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Wildlife survey relation:
Fragment 21 (Wildlife survey relation) = Wildlife survey ⟕ Fragment 1 (Forest relation)
Fragment 22 (Wildlife survey relation) = Wildlife survey ⟕ Fragment 2 (Forest relation)
Fragment 23 (Wildlife survey relation) = Wildlife survey ⟕ Fragment 3 (Forest relation)
Fragment 24 (Wildlife survey relation) = Wildlife survey ⟕ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Forest dispute registry relation:
Fragment 25 (Forest dispute registry) = Forest dispute registry ⟕ Fragment 1 (Forest relation)
Fragment 26 (Forest dispute registry) = Forest dispute registry ⟕ Fragment 2 (Forest relation)
Fragment 27 (Forest dispute registry) = Forest dispute registry ⟕ Fragment 3 (Forest relation)
Fragment 28 (Forest dispute registry) = Forest dispute registry ⟕ Fragment 4 (Forest relation)

Derived horizontal fragmentation of Forest dispute registry relation:
Fragment 29 (Fire relation) = Fire ⟕ Fragment 1 (Forest relation)
Fragment 30 (Fire relation) = Fire ⟕ Fragment 2 (Forest relation)
Fragment 31 (Fire relation) = Fire ⟕ Fragment 3 (Forest relation)
Fragment 32 (Fire relation) = Fire ⟕ Fragment 4 (Forest relation)

## 9.2 Vertical Fragmentation:

Divide a relation R into fragments R1, R2, .., Rr, each of which contains a subset of R's attributes as well as the primary key of R. Here the goal is to minimize the execution time of user applications that run on these fragments.
There are two types of design methodologies:
1. Top down
2. Bottom up

If there are too many alternatives for VF then use heuristic solutions based on:
1. Grouping: merge attributes to fragments
2. Splitting: divide a relation into fragments which will better for disjointness and easier dependency enforcement

Provided we have given a attribute usage matrix of a relation the vertical fragmentation can be done in 2 steps:

1. Cluster step: Permute rows and columns of the attribute affinity matrix to generate a clustered affinity matrix where attributes in each cluster are in high affinity to each other. BEA algorithm one of the algorithms used for this purpose.

**2.** Partition step: Divide the clustered attributes into non-overlapping partitions such that the number of application queries that access to more than one partition is as small as possible.

**Water relation:**

*Clustering step:*

Attribute affinity matrix of water relation:

| | A1 | A2 | A3 | A4 | A5 | A6 | A7 |
|---|---|---|---|---|---|---|---|
| Q1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q3 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| Q4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q8 | 1 | 1 | 0 | 0 | 0 | 1 | 1 |

| Query matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

Attribute affinity matrix:

| | A1 | A2 | A3 | A 4 | A5 | A6 | A7 |
|---|---|---|---|---|---|---|---|
| A1 | 100 | 100 | 0 | 0 | 0 | 100 | 100 |
| A2 | 0 | 170 | 0 | 0 | 0 | 170 | 170 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 100 | 170 | 0 | 0 | 0 | 170 | 170 |
| A7 | 100 | 170 | 0 | 0 | 0 | 170 | 170 |

Ordering the position of 3rd column A3 of Attribute Affinity MatrixM:

**Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+0-44000)= -88000

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(0+0-0)=0

Ordering the position of 4th column A4 of Attribute Affinity MatrixM:

**Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(0+0-44000)=-88000

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(0+0-0)=0

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

**Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(0+0-44000)=-88000

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(0+0-0)=0

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+44000-0)=88000

**Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(44000+96700 -44000) = 2x96700 = 193400**

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(96700+0-0)= 2 x 96700 = 193400

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))=2(0+96700-0)= 2 x 96700 = 193400

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+44000-0)=88000

**Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(44000+96700-44000)= 2 x 96700 = 193400**

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(96700+0-0)= 2 x 96700 = 193400

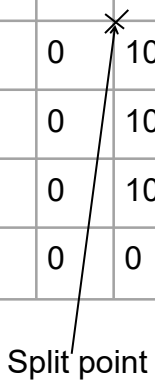
Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(0+96700-0)= 2 x 96700 = 193400

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(96700+0-0)= 193400

| Clustered Affinity Matrix | | | | | | | |
|---|---|---|---|---|---|---|---|
| | A3 | A4 | A5 | A1 | A6 | A7 | A2 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A1 | 0 | 0 | 0 | 100 | 100 | 100 | 100 |
| A6 | 0 | 0 | 0 | 100 | 170 | 170 | 170 |
| A7 | 0 | 0 | 0 | 100 | 170 | 170 | 170 |
| A2 | 0 | 0 | 0 | 0 | 170 | 170 | 170 |




TA - set of attributes in fragment f1 = A3,A4,A5

TB - set of attributes in fragment f2 = A6,A7,A2,A1

TQ - Number of applications accesses only TA = 0

BQ - Number of applications accesses only TB = Q3,Q8

OQ - Number of applications accesses both TA and TB = 0

CTQ - Total number of access to attributes by applications that access only TA = 0

CBQ - Total number of access to attributes by applications that access only TB = 170

COQ - Total number of access to attributes by applications that access both TA and TB = 0

Z=(CTQ*CBQ)-(COQ*COQ)

= (0 x 170) - (0x0) = 0

This one of the best partition based on BEA algorithm. The rest of the partitions either results to Z = 0 or Z < 0 so no point to include those calculations.

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A3,A4,A5) which will not be accessed by any query,perhaps I can separate that set out and create two fragments.

(A1, A2) is a key so it has to be present in both the fragments

Fragment A: A1,A2,A3,A4,A5
Fragment B: A6,A7,A2,A1

## Water Quality Relation

| Attribute usage matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
| Q1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q7 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| Q8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Query Matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

| Attribute Affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
| A1 | 100 | 100 | 0 | 0 | 0 | 0 | 0 | 100 | 0 |
| A2 | 100 | 100 | 0 | 0 | 0 | 0 | 0 | 100 | 0 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 100 | 100 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Ordering the position of 3rd column A3 of Attribute Affinity Matrix:

**Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+0-30000)=-60000

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(0+0-0)=0

Ordering the position of 4th column A4 of Attribute Affinity Matrix:

**Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(0+0-30000)=-60000

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(0+0-0)=0

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

**Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(0+0-30000)=-60000

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(0+0-0)=0

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

**Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(0+0-30000) = -60000

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))=2(0+0-0)= 0

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

**Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(0+0-30000)= -60000

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(0+0-0)=0

Ordering the position of 8th column A8 of Attribute Affinity Matrix:

**Cont(0,8,1)=2*(Bond(0,8)+Bond(8,1)-Bond(0,1))=2(0+20000-0)=40000**

Cont(1,8,2)=2*(Bond(1,8)+Bond(8,2)-Bond(1,2))=2(20000+20000-30000)= 20000

Cont(2,8,3)=2*(Bond(2,8)+Bond(8,3)-Bond(2,3))=2(20000+0-0)=40000

Cont(3,8,4)=2*(Bond(3,8)+Bond(8,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,8,5)=2*(Bond(4,8)+Bond(8,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,8,6)=2*(Bond(5,8)+Bond(8,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,8,7)=2*(Bond(6,8)+Bond(8,7)-Bond(6,7))=2(0+0-0)=0

Cont(7,8,9)=2*(Bond(7,8)+Bond(8,9)-Bond(7,9))=2(0+0-0)=0

Cont(9,8,10)=2*(Bond(9,8)+Bond(8,10)-Bond(9,10))=2(0+0-0)=0

Ordering the position of 9th column A8 of Attribute Affinity Matrix:

**Cont(0,9,1)=2*(Bond(0,9)+Bond(9,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,9,2)=2*(Bond(1,9)+Bond(9,2)-Bond(1,2))=2(0+0-30000)= -60000

Cont(2,9,3)=2*(Bond(2,9)+Bond(9,3)-Bond(2,3))=2(0+0-0)= 0

Cont(3,9,4)=2*(Bond(3,9)+Bond(9,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,9,5)=2*(Bond(4,9)+Bond(9,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,9,6)=2*(Bond(5,9)+Bond(9,6)-Bond(5,6))=2(0+0-0)=0

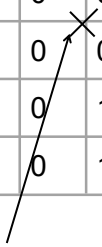
Cont(6,9,7)=2*(Bond(6,9)+Bond(9,7)-Bond(6,7))=2(0+0-0)=0

Cont(7,9,8)=2*(Bond(7,9)+Bond(9,8)-Bond(7,8))=2(0+0-0)=0

Cont(8,9,10)=2*(Bond(8,9)+Bond(9,10)-Bond(8,10))=2(0+0-0)=0

Water Quality

| Clustered Affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A3 | A4 | A5 | A6 | A7 | A9 | A8 | A1 | A2 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 100 |
| A1 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 100 | 100 |
| A2 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 100 | 100 |



Split point

**Partition Step:**

TA - set of attributes in fragment f1 = A3,A4,A5,A6,A7,A9

TB - set of attributes in fragment f2 = A8,A1,A2

TQ - Number of applications accesses only TA = 0

BQ - Number of applications accesses only TB = Q7

OQ - Number of applications accesses both TA and TB = 0

CTQ - Total number of access to attributes by applications that access only TA = 0

CBQ - Total number of access to attributes by applications that access only TB = 100

COQ - Total number of access to attributes by applications that access both TA and TB = 0

Z=(CTQ*CBQ)-(COQ*COQ) = (0 x 100) - (0x0) = 0

This one of the best partition based on BEA algorithm. The rest of the partitions either results to Z = 0 or Z < 0 so no point to include those calculations.

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A4,A5,A6,A7,A9) which will not be accessed by any query,perhaps I can separate that set out and create two fragments.

(A1, A2, A3) is a key so it has to be present in both the fragments

Fragment A: A1,A2,A3,A4,A5,A6,A7,A9
Fragment B: A1,A2,A3,A8

## Forest Relation:

| Attribute usage matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
| Q1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q2 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q3 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q4 | 1 | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 1 |
| Q5 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q6 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q7 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q8 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |

| Query Matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

| Attribute Affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
| A1 | 630 | 630 | 0 | 0 | 630 | 0 | 40 | 40 | 40 |
| A2 | 630 | 630 | 0 | 0 | 630 | 0 | 40 | 40 | 40 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 630 | 630 | 0 | 0 | 630 | 0 | 40 | 40 | 40 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 40 | 40 | 0 | 0 | 40 | 0 | 40 | 40 | 40 |
| A8 | 40 | 40 | 0 | 0 | 40 | 0 | 40 | 40 | 40 |
| A9 | 40 | 40 | 0 | 0 | 40 | 0 | 40 | 40 | 40 |

Ordering the position of 3rd column A3 of Attribute Affinity Matrix:

**Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+0-11,95,500)=-23,91,000

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(0+0-0)=0

Ordering the position of 4th column A4 of Attribute Affinity Matrix:

**Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(0+0-11,95,500)=-23,91,000

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(0+0-0)=0

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+11,95,500-0)=23,91,000

**Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(11,95,500+11,95,500-11,95,500)=23,91,000**

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(11,95,500+0-0)=23,91,000

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(0+0-0)=0

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

**Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(0+0-11,95,500)=-23,91,000

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))=2(0+0-80,400)= -1,60,800

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

**Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+80,400-0)= 1,60,800**

Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(80,400+80,400-11,95,500)= -20,69,400

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(80,400+0-0)=1,60,800

Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+80,400-0)=1,60,800

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(80,400+0-0)=1,60,800

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(0+9,600-0)=19,200

Ordering the position of 8th column A8 of Attribute Affinity Matrix:

Cont(0,8,1)=2*(Bond(0,8)+Bond(8,1)-Bond(0,1))=2(0+80,400-0)= 1,60,800

Cont(1,8,2)=2*(Bond(1,8)+Bond(8,2)-Bond(1,2))=2(80,400+80,400-11,95,500)= -20,69,400

**Cont(2,8,3)=2*(Bond(2,8)+Bond(8,3)-Bond(2,3))=2(80,400+0-0)=1,60,800**

Cont(3,8,4)=2*(Bond(3,8)+Bond(8,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,8,5)=2*(Bond(4,8)+Bond(8,5)-Bond(4,5))=2(0+80,400-0)=1,60,800

Cont(5,8,6)=2*(Bond(5,8)+Bond(8,6)-Bond(5,6))=2(0+80,400-0)=1,60,800

Cont(6,8,7)=2*(Bond(6,8)+Bond(8,7)-Bond(6,7))=2(0+9,600-0)=19,200

Cont(7,8,9)=2*(Bond(7,8)+Bond(8,9)-Bond(7,9))=2(9,600+9,600-9,600)=19,200

Cont(9,8,10)=2*(Bond(9,8)+Bond(8,10)-Bond(9,10))=2(9,600+0-0)=19,200

Ordering the position of 9th column A9 of Attribute Affinity Matrix:

Cont(0,9,1)=2*(Bond(0,9)+Bond(9,1)-Bond(0,1))=2(0+80,400-0)= 1,60,800

Cont(1,9,2)=2*(Bond(1,9)+Bond(9,2)-Bond(1,2))=2(80,400+80,400-11,95,500)= -20,69,400

**Cont(2,9,3)=2*(Bond(2,9)+Bond(9,3)-Bond(2,3))=2(80,400+0-0)=1,60,800**

Cont(3,9,4)=2*(Bond(3,9)+Bond(9,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,9,5)=2*(Bond(4,9)+Bond(9,5)-Bond(4,5))=2(0+80,400-0)=1,60,800

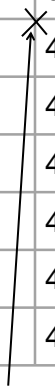
Cont(5,9,6)=2*(Bond(5,9)+Bond(9,6)-Bond(5,6))=2(0+80,400-0)=1,60,800

Cont(6,9,7)=2*(Bond(6,9)+Bond(9,7)-Bond(6,7))=2(0+9,600-0)=19,200

Cont(7,9,8)=2*(Bond(7,9)+Bond(9,8)-Bond(7,8))=2(9,600+9,600-9,600)=19,200

Cont(8,9,10)=2*(Bond(8,9)+Bond(9,10)-Bond(8,10))=2(9,600+0-0)=19,200

| Clustered affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A3 | A4 | A6 | A7 | A8 | A9 | A1 | A5 | A2 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 40 |
| A8 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 40 |
| A9 | 0 | 0 | 0 | 40 | 40 | 40 | 40 | 40 | 40 |
| A1 | 0 | 0 | 0 | 40 | 40 | 40 | 630 | 630 | 630 |
| A5 | 0 | 0 | 0 | 40 | 40 | 40 | 630 | 630 | 630 |
| A2 | 0 | 0 | 0 | 40 | 40 | 40 | 630 | 630 | 630 |



Split point

**Partition Step:**

TA - set of attributes in fragment f1 = A3, A4, A6

TB - set of attributes in fragment f2 = A7, A8, A9, A1, A5, A2

TQ - Number of applications accesses only TA = 0

BQ - Number of applications accesses only TB = Q1,Q2,Q3,Q4,Q5,Q6,Q7,Q8

OQ - Number of applications accesses both TA and TB

CTQ - Total number of access to attributes by applications that access only TA = 0

CBQ - Total number of access to attributes by applications that access only TB = 630

COQ - Total number of access to attributes by applications that access both TA and TB = 0

z= (0x630) - (0x0) = 0

This one of the best partition based on BEA algorithm. The rest of the partitions either results to Z = 0 or Z < 0 so no point to include those calculations.

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A3, A4, A6) which will not be accessed by any query,perhaps I can separate that set out and create two fragments.

(A1, A2, A3) is a key so it has to be present in both the fragments

Fragment A: A1,A2,A3,A5,A6,A7,A8,A9
Fragment B: A1,A2,A3,A4

## Wildlife Relation:

| | A1 | A2 | A3 | A5 | A6 | A7 | A8 | A9 |
|---|---|---|---|---|---|---|---|---|
| Q1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q2 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q4 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| Q5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q6 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q7 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q8 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Query Matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

| Attribute Affinity matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
| A1 | 400 | 0 | 0 | 290 | 110 | 0 | 0 | 0 |
| A2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 290 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 110 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Ordering the position of 3rd column A3 of Attribute Affinity Matrix:

**Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(0+0-0)=0

Ordering the position of 4th column A4 of Attribute Affinity Matrix:

Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0

**Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(1,16,000+0-0)= 2,32,000**

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(0+ 0-0)=0

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(0+31,900-0)=63,800

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+44,000-0) = 88,000

**Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(44,000+0-0)= 88,000**

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+31,900-0)= 63,800

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(31,900+0-0) = 63,800

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

**Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))=2(0+0-31,900)= -63,800

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))= 2(0+0-0)=0

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

**Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+0-0)= 0**

Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+0-31,900)= -63,800

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(0+0-0)=0

Ordering the position of 8th column A8 of Attribute Affinity Matrix:

**Cont(0,8,1)=2*(Bond(0,8)+Bond(8,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,8,2)=2*(Bond(1,8)+Bond(8,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,8,3)=2*(Bond(2,8)+Bond(8,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,8,4)=2*(Bond(3,8)+Bond(8,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,8,5)=2*(Bond(4,8)+Bond(8,5)-Bond(4,5))=2(0+0-31,900)= -63,800

Cont(5,8,6)=2*(Bond(5,8)+Bond(8,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,8,7)=2*(Bond(6,8)+Bond(8,7)-Bond(6,7))=2(0+0-0)=0

Cont(7,8,9)=2*(Bond(7,8)+Bond(8,9)-Bond(7,9))=2(0+0-0)=0

Cont(9,8,10)=2*(Bond(9,8)+Bond(8,10)-Bond(9,10))=2(0+0-0)=0

| Clustered affinity matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | A3 | A6 | A7 | A8 | A1 | A4 | A5 | A2 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A1 | 0 | 0 | 0 | 0 | 400 | 290 | 110 | 0 |
| A4 | 0 | 0 | 0 | 0 | 290 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 110 | 0 | 0 | 0 |
| A2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Split point

Partition step:

TA - set of attributes in fragment f1 = A3,A6,A7,A8

TB - set of attributes in fragment f2 = A1,A4,A5,A2

TQ - Number of applications accesses only TA = Q2,Q4,Q6,Q7,Q8

BQ - Number of applications accesses only TB = 0

OQ - Number of applications accesses both TA and TB = 0

CTQ - Total number of access to attributes by applications that access only TA = 400

CBQ - Total number of access to attributes by applications that access only TB = 0

COQ - Total number of access to attributes by applications that access both TA and TB = 0

Z= (0x400) - (0x0) = 0

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A3,A6,A7,A8) which will not be accessed by any query,perhaps I can separate that set out and create two fragments. (A1) is a key so it has to be present in both the fragments

Fragment A: A1,A3,A6,A7,A8
Fragment B: A1,A4,A5,A2

## Police Crime registry:

| Attribute usage matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A 7 | A8 | A9 |
| Q1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q5 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 |
| Q6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Query Matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

| Attribute Affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 | A9 |
| A1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A2 | 0 | 60 | 60 | 60 | 0 | 0 | 0 | 0 | 0 |
| A3 | 0 | 60 | 60 | 60 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 60 | 60 | 60 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Ordering the position of 3rd column A3 of Attribute Affinity Matrix:

Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0

**Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+10,800-0)=21,600**

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(10,800+10,800-10,800)=21,600

Ordering the position of 4th column A4 of Attribute Affinity Matrix:

Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0

**Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(0+10,800-0)= 21,600**

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(10,800+ 10,800-10,800)= 21,600

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(10,800 +0-0)= 21,600

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

**Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+0-0) = 0**

Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(0+0-0)= 0

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(0+0-10,800) = -21,600

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+0-10,800) = -21,600

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(0+0-0)=0

Maximum value of Cont is 22400. All functions producing the same value. Hence, we need not to change the position. Now order is CA(1,2,3,4,5)

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

**Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(0+0-10,800) = -21,600

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-10,800) = -21,600

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))= 2(0+0-0)=0

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))= 2(0+0-0)=0

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

**Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+0-0)= 0**

Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(0+0-10,800) = -21,600

Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-10,800) = -21,600

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(0+0-0)=0

Maximum value of Cont is 22400. Multiple functions are producing the same value. We can choose any one. Now order is CA(1,2,3,4,5,6,7)

Ordering the position of 8th column A8 of Attribute Affinity Matrix:

**Cont(0,8,1)=2*(Bond(0,8)+Bond(8,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,8,2)=2*(Bond(1,8)+Bond(8,2)-Bond(1,2))=2(0+0-0)=0

Cont(2,8,3)=2*(Bond(2,8)+Bond(8,3)-Bond(2,3))=2(0+0-10,800) = -21,600

Cont(3,8,4)=2*(Bond(3,8)+Bond(8,4)-Bond(3,4))=2(0+0-10,800) = -21,600

Cont(4,8,5)=2*(Bond(4,8)+Bond(8,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,8,6)=2*(Bond(5,8)+Bond(8,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,8,7)=2*(Bond(6,8)+Bond(8,7)-Bond(6,7))=2(0+0-0)=0

Cont(7,8,9)=2*(Bond(7,8)+Bond(8,9)-Bond(7,9))=2(0+0-0)=0

Cont(9,8,10)=2*(Bond(9,8)+Bond(8,10)-Bond(9,10))=2(0+0-0)=0

Police Crime registry

| Clustered affinity matrix | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | A5 | A6 | A7 | A8 | A9 | A1 | A3 | A4 | A2 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A9 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 60 | 60 | 60 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 60 | 60 | 60 |
| A2 | 0 | 0 | 0 | 0 | 0 | 0 | 60 | 60 | 60 |

split point

TA - set of attributes in fragment f1 = A1,A5,A6,A7,A8,A9

TB - set of attributes in fragment f2 = A2,A3,A4

TQ - Number of applications accesses only TA = 0

BQ - Number of applications accesses only TB = Q5

OQ - Number of applications accesses both TA and TB = 0

CTQ - Total number of access to attributes by applications that access only TA = 60

CBQ - Total number of access to attributes by applications that access only TB = 0

COQ - Total number of access to attributes by applications that access both TA and TB = 0

Z= (0x60) - (0x0) = 0

This one of the best partition based on BEA algorithm. The rest of the partitions either results to Z = 0 or Z < 0 so no point to include those calculations.

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A1,A5,A6,A7,A8,A9) which will not be accessed by any query,perhaps I can separate that set out and create two fragments. (A1, A2, A3) is a key so it has to be present in both the fragments

Fragment A: A1,A2,A3,A5,A6,A7,A8,A9
Fragment B: A1,A2,A3,A4

# Fire Department:

Attribute usage matrix

| | A1 | A2 | A3 | A4 | A6 | A7 | A8 | A9 |
|---|---|---|---|---|---|---|---|---|
| Q1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Q6 | 0 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| Q7 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 |
| Q8 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 |

| Query Matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 |
| Site 1 | 40 | 20 | 50 | 30 | 40 | 50 | 50 | 30 |
| Site 2 | 20 | 0 | 0 | 10 | 20 | 0 | 0 | 0 |
| Site 3 | 30 | 30 | 0 | 0 | 0 | 40 | 30 | 40 |
| Site 4 | 10 | 20 | 20 | 0 | 0 | 0 | 20 | 30 |
| Total | 100 | 70 | 70 | 40 | 60 | 90 | 100 | 100 |

Attribute affinity matrix

| | A1 | A2 | A3 | A4 | A5 | A6 | A7 | A8 |
|---|---|---|---|---|---|---|---|---|
| A1 | 200 | 200 | 0 | 0 | 0 | 0 | 100 | 100 |
| A2 | 200 | 290 | 0 | 0 | 0 | 0 | 190 | 100 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A7 | 100 | 190 | 0 | 0 | 0 | 0 | 0 | 0 |
| A8 | 100 | 100 | 0 | 0 | 0 | 0 | 0 | 0 |

Ordering the position of 3rd column A3 of Attribute Affinity Matrix:

**Cont(0,3,1)=2*(Bond(0,3)+Bond(3,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,3,2)=2*(Bond(1,3)+Bond(3,2)-Bond(1,2))=2(0+0-1,27,000)=-2,54,000

Cont(2,3,4)=2*(Bond(2,3)+Bond(3,4)-Bond(2,4))=2(0+0-0)=0

Ordering the position of 4th column A4 of Attribute Affinity Matrix:

**Cont(0,4,1)=2*(Bond(0,4)+Bond(4,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,4,2)=2*(Bond(1,4)+Bond(4,2)-Bond(1,2))=2(0+0-1,27,000)=-2,54,000

Cont(2,4,3)=2*(Bond(2,4)+Bond(4,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,4,5)=2*(Bond(3,4)+Bond(4,5)-Bond(3,5))=2(0+0-0)=0

Ordering the position of 5th column A5 of Attribute Affinity Matrix:

**Cont(0,5,1)=2*(Bond(0,5)+Bond(5,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,5,2)=2*(Bond(1,5)+Bond(5,2)-Bond(1,2))=2(0+0-1,27,000)=-2,54,000

Cont(2,5,3)=2*(Bond(2,5)+Bond(5,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,5,4)=2*(Bond(3,5)+Bond(5,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,5,6)=2*(Bond(4,5)+Bond(5,6)-Bond(4,6))=2(0+0-0)=0

Ordering the position of 6th column A6 of Attribute Affinity Matrix:

**Cont(0,6,1)=2*(Bond(0,6)+Bond(6,1)-Bond(0,1))=2(0+0-0)=0**

Cont(1,6,2)=2*(Bond(1,6)+Bond(6,2)-Bond(1,2))=2(0+0-1,27,000)=-2,54,000

Cont(2,6,3)=2*(Bond(2,6)+Bond(6,3)-Bond(2,3))=2(0+0-0)=0

Cont(3,6,4)=2*(Bond(3,6)+Bond(6,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,6,5)=2*(Bond(4,6)+Bond(6,5)-Bond(4,5))= 2(0+0-0)=0

Cont(5,6,7)=2*(Bond(5,6)+Bond(6,7)-Bond(5,7))= 2(0+0-0)=0

Ordering the position of 7th column A7 of Attribute Affinity Matrix:

**Cont(0,7,1)=2*(Bond(0,7)+Bond(7,1)-Bond(0,1))=2(0+58,000-0)= 1,16,000**

Cont(1,7,2)=2*(Bond(1,7)+Bond(7,2)-Bond(1,2))=2(58,000+58,000-1,27,000)= -22,000

Cont(2,7,3)=2*(Bond(2,7)+Bond(7,3)-Bond(2,3))=2(58,000+0-0) = 1,16,000

Cont(3,7,4)=2*(Bond(3,7)+Bond(7,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,7,5)=2*(Bond(4,7)+Bond(7,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,7,6)=2*(Bond(5,7)+Bond(7,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,7,8)=2*(Bond(6,7)+Bond(7,8)-Bond(6,8))=2(0+29,000-0)= 58,000

Ordering the position of 8th column A8 of Attribute Affinity Matrix:

Cont(0,8,1)=2*(Bond(0,8)+Bond(8,1)-Bond(0,1))=2(0+40000-0)=80000

Cont(1,8,2)=2*(Bond(1,8)+Bond(8,2)-Bond(1,2))=2(40000+49000-1,27,000)=-38000

**Cont(2,8,3)=2*(Bond(2,8)+Bond(8,3)-Bond(2,3))=2(49000+0-0) = 98000**

Cont(3,8,4)=2*(Bond(3,8)+Bond(8,4)-Bond(3,4))=2(0+0-0)=0

Cont(4,8,5)=2*(Bond(4,8)+Bond(8,5)-Bond(4,5))=2(0+0-0)=0

Cont(5,8,6)=2*(Bond(5,8)+Bond(8,6)-Bond(5,6))=2(0+0-0)=0

Cont(6,8,7)=2*(Bond(6,8)+Bond(8,7)-Bond(6,7))=2(0+29,000-0)= 58,000

Cont(7,8,9)=2*(Bond(7,8)+Bond(8,9)-Bond(7,9))=2(29,000+0-0)=58,000

| Clustered affinity matrix | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | A3 | A4 | A5 | A6 | A8 | A7 | A1 | A2 |
| A3 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A4 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A1 | 0 | 0 | 0 | 0 | 100 | 100 | 200 | 200 |
| A2 | 0 | 0 | 0 | 0 | 100 | 190 | 200 | 290 |
| A7 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 190 |
| A8 | 0 | 0 | 0 | 0 | 0 | 0 | 100 | 100 |

Split point

*Partitioning Step:*

TA - set of attributes in fragment f1 = A3,A4,A5,A6

TB - set of attributes in fragment f2 = A1,A2,A7,A8

TQ - Number of applications accesses only TA = 0

BQ - Number of applications accesses only TB = Q5

OQ - Number of applications accesses both TA and TB = 0

CTQ - Total number of access to attributes by applications that access only TA = 0

CBQ - Total number of access to attributes by applications that access only TB = 290

COQ - Total number of access to attributes by applications that access both TA and TB = 0

Z= (0x290) - (0x0) = 0

This one of the best partition based on BEA algorithm. The rest of the partitions either results to Z = 0 or Z < 0 so no point to include those calculations.

As Z = 0 there is no benefit in vertical partitioning as per BEA algorithm. Considering there is a set of attributes (A3,A4,A5,A6) which will not be accessed by any query,perhaps I can separate that set out and create two fragments. (A1, A2) is a key so it has to be present in both the fragments

Fragment A: A1,A2,A3,A4,A5,A6
Fragment B: A1,A2,A7,A8

# 10. PHYSICAL DESIGN:

This physical design talks about how these fragments are stores in secondary memory. Based on global schema defined in Section 6, the size of all the attributes of all relations remains same. Following are the assumptions which are considered for the physical design:

- Fixed length records are considered for all relations.
- The delimiter for each field is length of the field
- Total number of records in respective relations (provided in below table).
- Block size is 1024 bytes.
- Record doesn't span over multiple blocks (this can be achieved by taking floor function during calculating number of records per block to restrict single record doesn't span over blocks).
- Block pointer(Bp) size is 4 bytes
- Average Seek Time(S) is 10 ms irrespective of any site.
- Average Disk rotation time (Latency) Time (L) is 9 ms irrespective of any site.
- Block transfer rate (Tr) is 1 ms irrespective of any site.

| Fragment | Relation | # of records | Record size in bytes | Blocking factor | # no of blocks |
|---|---|---|---|---|---|
| Fragment 1 | Forest Department | 25 | 44 | 1024/44 = 23 | 2 |
| Fragment 2 | Forest Department | 20 | 44 | 23 | 1 |
| Fragment 3 | Forest Department | 30 | 44 | 23 | 2 |
| Fragment 4 | Forest Department | 25 | 44 | 23 | 2 |
| Fragment 5 | Water Department | 25 | 38 | 1024/38 = 26 | 1 |
| Fragment 6 | Water Department | 20 | 38 | 26 | 1 |
| Fragment 7 | Water Department | 30 | 38 | 26 | 2 |
| Fragment 8 | Water Department | 25 | 38 | 26 | 1 |
| Fragment 9 | Water Quality | 100 | 24 | 1024/24 = 42 | 3 |
| Fragment 10 | Water Quality | 80 | 24 | 42 | 2 |
| Fragment 11 | Water Quality | 120 | 24 | 42 | 3 |
| Fragment 12 | Water Quality | 100 | 24 | 42 | 3 |
| Fragment 13 | Police Department | 125 | 28 | 1024/28 = 36 | 4 |
| Fragment 14 | Police Department | 100 | 28 | 36 | 3 |
| Fragment 15 | Police Department | 150 | 28 | 36 | 5 |
| Fragment 16 | Police Department | 125 | 28 | 36 | 4 |
| Fragment 17 | Police Crime registry | 1000 | 40 | 1024/40 = 25 | 40 |
| Fragment 18 | Police Crime registry | 800 | 40 | 25 | 20 |

| | | | | | |
|---|---|---|---|---|---|
| Fragment 19 | Police Crime registry | 1200 | 40 | 25 | 60 |
| Fragment 20 | Police Crime registry | 1000 | 40 | 25 | 40 |
| Fragment 21 | Wildlife Survey | 25 | 16 | 1024/16 = 64 | 1 |
| Fragment 22 | Wildlife Survey | 20 | 16 | 64 | 1 |
| Fragment 23 | Wildlife Survey | 30 | 16 | 64 | 1 |
| Fragment 24 | Wildlife Survey | 25 | 16 | 64 | 1 |
| Fragment 25 | Forest dispute registry | 100 | 18 | 1024/18 = 56 | 2 |
| Fragment 26 | Forest dispute registry | 80 | 18 | 56 | 2 |
| Fragment 27 | Forest dispute registry | 120 | 18 | 56 | 3 |
| Fragment 28 | Forest dispute registry | 100 | 18 | 56 | 2 |
| Fragment 29 | Fire department | 125 | 34 | 1024/34 = 30 | 5 |
| Fragment 30 | Fire department | 100 | 34 | 30 | 4 |
| Fragment 31 | Fire department | 150 | 34 | 30 | 5 |
| Fragment 32 | Fire department | 125 | 34 | 30 | 5 |

Types of Queries and preferred access methods:

**Query type 1:** access all records of a given type
“Increase everyone’s salary by 10%”
Preferred access method: *sequential processing*

**Query type 2:** access at most one record
“Find the address of John Smith,
whose id number is 333-44-5555”
Preferred access methods: *hashing, B+ tree index*

**Query type 3:** access a subset of records of a given type
“Find all employees who have C programming experience and over three years with the company”
Preferred access method: *secondary indexing*

**Access Time to local query:**

Next we will calculate the time taken to query each relation considering the relation is locally present. Even though in our sample SQL’s are just read still provided the Local (keyword local because it’s not yet distributed) Query Time and Local Update Time formulae

(i) **Local Query Time** = (Seek Time + Latency + Block Transfer Time) * N
(ii) **Local Update Time** = (Seek Time + Latency + Block Transfer Time) * N * 2

Where,

- N is number of disk block access, which depends on the relation (we already calculated this above and will consider indexed logic # of block access).
- *2 is included in the Update time, since the data block has to be fetched into memory from the disk, updated and then written back to the disk

**Access Time to Remote query:**

Let us consider the distance between sites. Assume that each site is located at some distance say 100kms from the other site and the speed of the transmission media connecting the sites is 10^7 meters/second. Propagation delay between the sites is computed as below.

(i) **Propagation Delay** = (Distance between sites)/(Speed of Transmission media) = 100 * 10^3 / 2 * 10^6 which will be = 0.05s = 50 ms

Let us assume bandwidth of the network as 1MBps and data is exchanged between sites in form of packets. Package size is assumed to be 1500bytes. Transmission Time for a packet is given by,

(ii) **Packet Transmission Time** = (Size of packet) / Bandwidth = (1500 B) / (10^6 B/s) = 0.0015s, approximately equal to 2ms
(iii) **Remote Query Time** = Local Query Time + 2 * Propagation Delay + Packet Transmission Time
(iv) **Remote Update Time** = Local Update Time + 2 * Propagation Delay

Packet Transmission Time is included in Remote Query Time because; the result of the query will contain some data which is not negligible. But this data depends on the query, so it is assumed to be one packet, on an average. Using the formulations made above, the below table can be constructed.

Lets consider

A = # of block access With Indexing
B = Local Query Time (ms) = (S+L+Tr)
C = Remote Query time

| Fragment | Relation | # no of blocks | Access method | A | B | C |
|---|---|---|---|---|---|---|
| Fragment 1 | Forest Department | 2 | Sequential | - | 40 | 142 |
| Fragment 2 | Forest Department | 1 | Sequential | - | 20 | 122 |
| Fragment 3 | Forest Department | 2 | Sequential | - | 40 | 142 |
| Fragment 4 | Forest Department | 2 | Sequential | - | 40 | 142 |
| Fragment 5 | Water Department | 1 | Sequential | - | 20 | 122 |
| Fragment 6 | Water Department | 1 | Sequential | - | 20 | 122 |
| Fragment 7 | Water Department | 2 | Sequential | - | 40 | 142 |
| Fragment 8 | Water Department | 1 | Sequential | - | 20 | 122 |
| Fragment 9 | Water Quality | 3 | Clustered Index | 2 | 40 | 142 |
| Fragment 10 | Water Quality | 2 | Sequential | - | 40 | 142 |
| Fragment 11 | Water Quality | 3 | Clustered Index | 2 | 40 | 142 |
| Fragment 12 | Water Quality | 3 | Clustered Index | 2 | 40 | 142 |
| Fragment 13 | Police Department | 4 | Clustered Index | 2 | 40 | 142 |
| Fragment 14 | Police Department | 3 | Clustered Index | 2 | 40 | 142 |
| Fragment 15 | Police Department | 5 | Clustered Index | 2 | 40 | 142 |
| Fragment 16 | Police Department | 4 | Clustered Index | 2 | 40 | 142 |
| Fragment 17 | Police Crime registry | 40 | Clustered Index | 2 | 40 | 142 |
| Fragment 18 | Police Crime registry | 20 | Clustered Index | 2 | 40 | 142 |
| Fragment 19 | Police Crime registry | 60 | Clustered Index | 2 | 40 | 142 |
| Fragment 20 | Police Crime registry | 40 | Clustered Index | 2 | 40 | 142 |
| Fragment 21 | Wildlife Survey | 1 | Sequential | - | 20 | 122 |
| Fragment 22 | Wildlife Survey | 1 | Sequential | - | 20 | 122 |
| Fragment 23 | Wildlife Survey | 1 | Sequential | - | 20 | 122 |
| Fragment 24 | Wildlife Survey | 1 | Sequential | - | 20 | 122 |
| Fragment 25 | Forest dispute registry | 2 | Sequential | - | 40 | 142 |
| Fragment 26 | Forest dispute registry | 2 | Sequential | - | 40 | 142 |
| Fragment 27 | Forest dispute registry | 3 | Clustered Index | 2 | 40 | 142 |
| Fragment 28 | Forest dispute registry | 2 | Sequential | - | 40 | 142 |
| Fragment 29 | Fire department | 5 | Clustered Index | 2 | 40 | 142 |
| Fragment 30 | Fire department | 4 | Clustered Index | 2 | 40 | 142 |
| Fragment 31 | Fire department | 5 | Clustered Index | 2 | 40 | 142 |
| Fragment 32 | Fire department | 5 | Clustered Index | 2 | 40 | 142 |

# 11. ALLOCATION AND REPLICATION:

For allocating the fragments to sites we considered the **Redundant All Beneficial Sites method**. Transaction table is given below: consider there are four sites: S1, S2, S3, S4

| Transaction | Originating sites | frequency | Fragment Access |
|---|---|---|---|
| Q1 | S1, S2 | 200 | F1 – 25 reads<br>F2 – 20 reads<br>F3 – 30 reads<br>F4 – 25 reads |
| Q2 | S1, S4 | 250 | F1 – 25 reads, F21 - 25 reads<br>F2 – 20 reads, F22 – 20 reads<br>F3 – 30 reads, F23 – 30 reads<br>F4 – 25 reads, F24 – 25 reads |
| Q3 | S2, S3 | 200 | F1 – 25 reads, F5 - 100 reads<br>F2 – 20 reads, F6 – 80 reads<br>F3 – 30 reads, F7 – 120 reads<br>F4 – 25 reads, F8 – 100 reads |
| Q4 | S1, S2, S3 | 400 | F1 – 25 reads, F21 - 25 reads<br>F2 – 20 reads, F22 – 20 reads<br>F3 – 30 reads, F23 – 30 reads<br>F4 – 25 reads, F24 – 25 reads |
| Q5 | S1, S2, S4 | 150 | F1 – 25 reads, F21 - 1000 reads<br>F2 – 20 reads, F22 – 800 reads<br>F3 – 30 reads, F23 – 1200 reads<br>F4 – 25 reads, F24 – 1000 reads |

| | | | |
|---|---|---|---|
| Q6 | S1,S4 | 250 | F1 – 25 reads, F21 - 25 reads<br>F2 – 20 reads, F22 – 800 reads<br>F3 – 30 reads, F23 – 1200 reads<br>F4 – 25 reads, F24 – 1000 reads |
| Q7 | S3, S4 | 400 | F1 - 25 reads, F9 - 100 reads, F29 - 125 reads, F21 - 25 reads<br>F2 - 20 reads, F10 - 80 reads, F30 - 100 reads, F22 - 20 reads<br>F3 - 30 reads, F11 - 120 reads, F31 - 150 reads, F23 - 30 reads<br>F4 - 25 reads, F12 - 100 reads, F32 - 125 reads, F24 - 25 reads |
| Q8 | S1, S2 | 300 | F1 - 25 reads, F5 - 100 reads, F29 - 125 reads, F21 - 25 reads<br>F2 - 20 reads, F6 - 80 reads, F30 - 100 reads, F22 - 20 reads<br>F3 - 30 reads, F7 - 120 reads, F31 - 150 reads, F23 - 30 reads<br>F4 - 25 reads, F8 - 100 reads, F32 - 125 reads, F24 - 25 reads |

Since our sample queries are related only read (not update) will calculate only Benefit Computation (not Cost computation) of placing a fragment at a particular site. Let us proceed to Benefit Computation. Benefit computation is based on read queries. The benefit of placing each fragment at each site is given in the below table.

| Fragments | Originating sites | Read query | # of reads * Freq * (Remote Time - Local Time) | Benefit(ms) |
|---|---|---|---|---|
| F1 | S1 | Q1,Q2,Q4,Q5,Q6,Q8 | 25 x (200 +250+400+150+250+300 ) x (142-40) | **39,52,500** |
| - | S2 | Q1,Q3,Q4,Q5,Q8 | 25 x (200+200+400+150+300 ) x (122-20) | 31,87,500 |
| - | S3 | Q3,Q4 | 25 x (200+400) x (142-40) | 15,30,000 |
| - | S4 | Q2,Q5,Q6,Q7 | 25 x (250+150+250+400 ) x (142-40) | 26,77,500 |
| F2 | S1 | Q1,Q2,Q4,Q5,Q6,Q8 | 20 x (200 +250+400+150+250+300 ) x (122-20) | **31,62,000** |
| - | S2 | Q1,Q3,Q4,Q5,Q8 | 20 x (200+200+400+150+300 ) x (122-20) | 25,50,000 |
| - | S3 | Q3,Q4 | 20 x (200+400) x (122-20) | 12,24,000 |
| - | S4 | Q2,Q5,Q6,Q7 | 20 x (250+150+250+400 ) x (122-20) = | 21,42,000 |
| F3 | S1 | Q1,Q2,Q4,Q5,Q6,Q8 | 30 x (200 +250+400+150+250+300 ) x (142-40) | **47,43,000** |
| - | S2 | Q1,Q3,Q4,Q5,Q8 | 30 x (200+200+400+150+300 ) x (142-40) | 38,25,000 |
| - | S3 | Q3,Q4 | 30 x (200+400) x (142-40) | 18,36,000 |
| - | S4 | Q2,Q5,Q6,Q7 | 30 x (250+150+250+400 ) x (142-40) | 32,13,000 |
| F4 | S1 | Q1,Q2,Q4,Q5,Q6,Q8 | 25 x (200 +250+400+150+250+300 ) x (142-40) | **39,52,500** |

| | | | | |
|---|---|---|---|---|
| - | S2 | Q1,Q3,Q4,Q5,Q8 | 25 x (200+200+400+150+300 ) x (142-40) | 31,87,500 |
| - | S3 | Q3,Q4 | 25 x (200+400) x (142-40) | 15,30,000 |
| - | S4 | Q2,Q5,Q6,Q7 | 25 x (250+150+250+400 ) x (142-40) | 26,77,500 |
| F5 | S1 | Q8 | 25 x (300 ) x (122-20) | 7,65,000 |
| - | S2 | Q8,Q3 | 25 x (200+300 ) x (122-20) | **12,75,000** |
| - | S3 | Q3 | 25 x (200) x (122-20) | 5,10,000 |
| - | S4 | - | 0 | 0 |
| F6 | S1 | Q8 | 20 x (300 ) x (122-20) | 6,12,000 |
| - | S2 | Q8,Q3 | 20 x (200+300 ) x (122-20) | **10,20,000** |
| - | S3 | Q3 | 20 x (200) x (122-20) | 4,08,000 |
| - | S4 | - | 0 | |
| F7 | S1 | Q8 | 30 x (300 ) x (142-40) | 9,18,000 |
| - | S2 | Q8,Q3 | 30 x (200+300 ) x (142-40) | **15,30,000** |
| - | S3 | Q3 | 30 x (200) x (142-40) | 6,12,000 |
| - | S4 | - | 0 | 0 |
| F8 | S1 | Q8 | 25 x (300 ) x (122-20) | 7,65,000 |
| - | S2 | Q8,Q3 | 25 x (200+300 ) x (122-20) | **12,75,000** |
| - | S3 | Q3 | 25 x (200) x (122-20) | 5,10,000 |

| | | | | |
|---|---|---|---|---|
| _ | S4 | - | 0 | 0 |
| F9 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | Q7 | 100 x ( 400 ) x (142-40) | **40,80,000** |
| _ | S4 | Q7 | 100 x (400 ) x (142-40) | 40,80,000 |
| F10 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | Q7 | 80 x ( 400 ) x (142-40) | **32,64,000** |
| _ | S4 | Q7 | 80 x (400 ) x (142-40) | 32,64,000 |
| F11 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | Q7 | 120 x (400 ) x (142-40) | **48,96,000** |
| _ | S4 | Q7 | 120 x (400 ) x (142-40) | 48,96,000 |
| F12 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | Q7 | 100 x (400 ) x (142-40) | **40,80,000** |
| _ | S4 | Q7 | 100 x (400 ) x (142-40) | 40,80,000 |
| F13 | S1 | - | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |
| - | S4 | - | 0 | 0 |
| F14 | S1 | - | 0 | 0 |
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |
| - | S4 | - | 0 | 0 |
| F15 | S1 | - | 0 | 0 |
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |
| - | S4 | - | 0 | 0 |
| F16 | S1 | - | 0 | 0 |
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |
| - | S4 | - | 0 | 0 |
| F17 | S1 | Q5 | 1000 x (150 ) x (142-40) | **1,53,00,000** |
| - | S2 | Q5 | 1000 x (150 ) x (142-40) | 1,53,00,000 |
| - | S3 | - | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| _ | S4 | Q5 | 1000 x (150 ) x (142-40) | 1,53,00,000 |
| F18 | S1 | Q5 | 800 x (150 ) x (142-40) | **1,22,40,000** |
| _ | S2 | Q5 | 800 x (150 ) x (142-40) | 1,22,40,000 |
| _ | S3 | - | 0 | 0 |
| _ | S4 | Q5 | 800 x (150 ) x (142-40) | 1,22,40,000 |
| F19 | S1 | Q5 | 1200 x (150 ) x (142-40) | **1,83,60,000** |
| _ | S2 | Q5 | 1200 x (150 ) x (142-40) | 1,83,60,000 |
| _ | S3 | - | 0 | 0 |
| _ | S4 | Q5 | 1200 x (150 ) x (142-40) | 1,83,60,000 |
| F20 | S1 | Q5 | 1000 x (150 ) x (142-40) | **1,53,00,000** |
| _ | S2 | Q5 | 1000 x (150 ) x (142-40) | 1,53,00,000 |
| _ | S3 | - | 0 | 0 |
| _ | S4 | Q5 | 1000 x (150 ) x (142-40) | 1,53,00,000 |
| F21 | S1 | Q2 Q4 Q6 Q8 | 25 x (250+ 400 +250+300 ) x (122-20) | **30,60,000** |
| _ | S2 | Q4, Q8 | 25 x (400+300 ) x (122-20) | 17,85,000 |
| _ | S3 | Q4,Q7 | 25 x (400+ 400) x (122-20) | 20,40,000 |
| _ | S4 | Q2, Q6,Q7 | 25 x (250+250+ 400) x (122-20) | 22,95,000 |
| F22 | S1 | Q2 Q4 Q6 Q8 | 20 x (250+ 400 +250+300 ) x (122-20) | **24,48,000** |

| | | | | |
|---|---|---|---|---|
| - | S2 | Q4, Q8 | 20 x (400+300 ) x (122-20) | 14,28,000 |
| - | S3 | Q4,Q7 | 20 x (400+300 ) x (122-20) | 14,28,000 |
| - | S4 | Q2, Q6,Q7 | 20 x(250+250+ 400) x (122-20) | 18,36,000 |
| F23 | S1 | Q2 Q4 Q6 Q8 | 30 x (250+ 400 +250+300 ) x (122-20) | **36,72,000** |
| - | S2 | Q4, Q8 | 30 x (400+300 ) x (122-20) | 21,42,000 |
| - | S3 | Q4,Q7 | 30 x (400+300 ) x (122-20) | 21,42,000 |
| - | S4 | Q2, Q6,Q7 | 30 x (250+250+ 400) x (122-20) | 27,54,000 |
| F24 | S1 | Q2 Q4 Q6 Q8 | 25 x (250+ 400 +250+300 ) x (122-20) | **30,60,000** |
| - | S2 | Q4, Q8 | 25 x (400+300 ) x (122-20) | 17,85,000 |
| - | S3 | Q4,Q7 | 25 x (400+300 ) x (122-20) | 20,40,000 |
| - | S4 | Q2, Q6,Q7 | 25 x (250+250+ 400) x (122-20) | 22,95,000 |
| F25 | S1 | - | 0 | 0 |
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |
| - | S4 | - | 0 | 0 |
| F26 | S1 | - | 0 | 0 |
| - | S2 | - | 0 | 0 |
| - | S3 | - | 0 | 0 |

| | | | | |
|---|---|---|---|---|
| _ | S4 | - | 0 | 0 |
| F27 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | - | 0 | 0 |
| _ | S4 | - | 0 | 0 |
| F28 | S1 | - | 0 | 0 |
| _ | S2 | - | 0 | 0 |
| _ | S3 | - | 0 | 0 |
| _ | S4 | - | 0 | 0 |
| F29 | S1 | Q6, Q8 | 125 x (250 + 300 ) x (142-40) | 70,12,500 |
| _ | S2 | Q8 | 125 x (300 ) x (142-40) | 38,25,000 |
| _ | S3 | Q7 | 125 x (400 ) x (142-40) | 51,00,000 |
| _ | S4 | Q6,Q7 | 125 x (250+ 400) x (142-40) | **82,87,500** |
| F30 | S1 | Q6, Q8 | 100 x (250 + 300 ) x (142-40) | 56,10,000 |
| _ | S2 | Q8 | 100 x (300 ) x (142-40) | 30,60,000 |
| _ | S3 | Q7 | 100 x (400 ) x (142-40) | 40,80,000 |
| _ | S4 | Q6,Q7 | 100 x (250+ 400) x (142-40) | **66,30,000** |
| F31 | S1 | Q6, Q8 | 150 x (250 + 300 ) x (142-40) | 84,15,000 |

| | | | | |
|---|---|---|---|---|
| - | S2 | Q8 | 150 x (300 ) x (142-40) | 45,90,000 |
| - | S3 | Q7 | 150 x (400 ) x (142-40) | 61,20,000 |
| - | S4 | Q6,Q7 | 150 x (250+ 400) x (142-40) | **99,45,000** |
| F32 | S1 | Q6, Q8 | 125 x (250 + 300 ) x (142-40) | 70,12,500 |
| - | S2 | Q8 | 125 x (300 ) x (142-40) | 38,25,000 |
| - | S3 | Q7 | 125 x (400 ) x (142-40) | 51,00,000 |
| - | S4 | Q6,Q7 | 125 x (250+ 400) x (142-40) | **82,87,500** |

Since the cost of placing each fragment at each site is 0, the Benefit computation which is done it-self gives the Benefit – Cost value. So based on the Benefit table, we can infer the following:

Fragment1: is assigned at S1 and replicated to S2,S3,S4
Fragment2: is assigned at S1 and replicated to S2,S3,S4
Fragment3: is assigned at S1 and replicated to S2,S3,S4
Fragment4: is assigned at S1 and replicated to S2,S3,S4
Fragment5: is assigned at S2 and replicated to S1,S3
Fragment6: is assigned at S2 and replicated to S1,S3
Fragment7: is assigned at S2 and replicated to S1,S3
Fragment8: is assigned at S2 and replicated to S1,S3
Fragment9: is assigned at S3 and replicated to S4
Fragment10: is assigned at S3 and replicated to S4
Fragment11: is assigned at S3 and replicated to S4
Fragment12: is assigned at S3 and replicated to S4
Fragment17: is assigned at S1 and replicated to S2, S4
Fragment18: is assigned at S1 and replicated to S2, S4
Fragment19: is assigned at S1 and replicated to S2, S4
Fragment20: is assigned at S1 and replicated to S2, S4
Fragment21: is assigned at S1 and replicated to S2,S3,S4
Fragment22: is assigned at S1 and replicated to S2,S3,S4
Fragment23: is assigned at S1 and replicated to S2,S3,S4

Fragment24: is assigned at S1 and replicated to S2,S3,S4

Fragment29: is assigned at S4 and replicated to S2,S3,S1
Fragment30: is assigned at S4 and replicated to S2,S3,S1
Fragment31: is assigned at S4 and replicated to S2,S3,S1
Fragment32: is assigned at S4 and replicated to S2,S3,S1