REPORT PHASE-1 - QuarantinedAgain

Team Members:

Manas Kabre - 2018111014
Nikunj Nawal - 2018111011

Files:

- Images are named accordingly and have the Application ER Diagram, Application Relational Model, System Catalogue ER Diagram and System Catalogue Relational Model.
- All the sql files in the Sql_Files folder contain the query used to create and populate the system catalogue and also contain the queries used to create the Application database schema fragments at different sites.

Sql_Files/<file>	Function
Syscat.sql	Queries used to create the System Catalogue tables
Application1.sql	Creates all the fragments of tables with suffix 1
Application2.sql	Creates all the fragments of tables with suffix 2
Application3.sql	Creates all the fragments of tables with suffix 3
Sites.sql	Populates the Sites table of the System Catalogue
Tables.sql	Populates the Tables table of the System Catalogue
Frag_Table.sql	Populates the Frag_Table table of the System Catalogue
Allocation.sql	Populates the Allocation table of the System Catalogue
Columns.sql	Populates the Columns table of the System Catalogue

- In the sites allotted to use, fragments of different tables have a suffix of their number (1,2,3).
- On all sites there is a Database name QuarantinedAgain and there we have all the system catalogue tables replicated and the fragmented tables are placed according to their allocation schema and created according to their fragmentation schema.

APPLICATION FIELD OR MINI-WORLD :

- Course Registration Platform of A College

There are three types of entities - “Student” , “Faculty” and “Course” where every course can be taught by one or more faculty and each faculty can teach any number of courses (including no course). Every student can opt for any number of courses (including no course). All the entities are strong entities. The attributes of each entity and relations are present in ER Model (Application ER Diagram.png) or relational model (Application Relational model.png).

System Catalogue of The Application DBMS :

There are 3 strong entities namely “Sites”, “Tables” and “Frag_Table” and a weak entity “Columns” of “Tables” as the identifying entity. Each entry in Tables is related to one or many entries of “Frag_Table” using the relationship “Fragmented” and it is mandatory for both the tables. Each entry of “Frag_Table” is related to exactly 2 sites using relationship “Allocation” whereas a site can be related to any number of fragments. The attributes of all entities are present in ER Model (System Catalogue ER Diagram.png) or relational model (System Catalogue Relational Model.png).

Fragmentation Schema Explanation :

- Horizontal Fragmentation :

  The “Course” table is horizontally fragmented on the Course_Type attribute which can take 3 values CSE, ECE, HSME. These define the type of course. This fragmentation will help in the following queries.

  ```
  SELECT Course_Name FROM Course WHERE Course_Type = “CSE”
  ```

  And the Course_Type can be changed as required.

- Vertical Fragmentation :

Student and Faculty tables are vertically fragmented. Groups of Columns are maintained in different sites. This is will in queries where few columns are to be selected and not all.

SELECT A,B,C FROM T WHERE <Condition>

Where A,B,C are columns which belong to the same fragment and condition involves columns in the same fragment too,

- Derived Horizontal Fragmentation

Opts, Teaches and Course_Prerequisites are tables which have derived horizontal fragmentation. These are derived from the Course table horizontal fragmentation. These are helpful in queries similar to Horizontal fragmentation.

```
SELECT Faculty.First_Name FROM Faculty,Teaches,Course WHERE Teaches.Faulty_Id = Faculty.Faculty_Id AND Course.Course_Type = “CSE”
```

And similar such queries will be executed faster and more efficiently. Since we don't have to perform a join with Course, as we know there is horizontal fragmentation on Course_Type and Teaches has derived horizontally, hence we can just directly go to the appropriate site and use all the entries there.

Allocation Schema:

We have given each fragment two sites.

Table_Name	Fragmentation_Type	Fragment1 (sites)	Fragment2 (sites)	Fragment3 (sites)
Student	Vertical	1,2	2,3	3,1
Course	Horizontal	1,2	2,3	3,1
Course_Prerequisites	Derived Horizontal	1,2	2,3	3,1
Opts	Derived Horizontal	1,2	2,3	3,1
Teaches	Derived Horizontal	1,2	2,3	3,1
Faculty	Vertical	1,2	2,3	3,1

We have given two fragments to each table in order to have some backup in case of any failure in future.

The horizontal and derived horizontal fragmentation will have their respective fragments in the same site.

Code :

- The file is phase_1.py.
- Run the code on 10.3.5.215. The code is written to be run on that.
- It has 2 functions :
- Get_Fragments() prints a list of the fragments which are available along with their names and conditions.
- Get_Allocation() which takes fragment id as the input and outputs the sites on which the fragment exists.