REPORT - QuarantinedAgain

Team Members:

Manas Kabre - 2018111014
Nikunj Nawal - 2018111011

Files:

- Images are named accordingly and have the Application ER, Application Relational model, System_Catalogue ER 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.

Sql_Files/<file>	Function
system_cat.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
table_info.sql	Populates the table_info table of the System Catalogue
fragmentted_table.sql	Populates the fragmentted_table table of the System Catalogue
allocation.sql	Populates the Allocation table of the System Catalogue
column_sql	Populates the column_info 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_info” , “Faculty” and “Course_Offered” 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). A student can also pursue any number of “BTP(B.Tech Project) / IS(Independent Study) / Honours Project” but the restriction is only one out of these three can be pursued under a particular faculty. All the entities are strong entities. The attributes of each entity and relations are present in ER Model (Application ER.png) or relational model (Application Relational model.png).

System Catalogue of The Application DBMS :

There are 3 strong entities namely “Sites”, “Table_info” and “fragmentted_table” and a weak entity “column_info” of “Table_info” as the identifying entity. Each entry in Table_info is related to one or many entries of “fragmentted_table” using relationship “fragmented” and it is mandatory for both the tables. Each entry of “fragmentted_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.png) or relational model (System Catalogue Relational Model.png).

Fragmentation Schema Explanation :

- ### Horizontal Fragmentation :

  Course Offered 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_Offered WHERE Course_Type = “CSE”
```

And the Course_Type can be changed as required.

BTP_IS_Hon Is also horizontally fragmented. It is fragmented on the column type which can take 3 values BTP, Hon,IS. This fragmentation helps in the following queries

```
SELECT COUNT(Grade) FROM BTP_IS_Hon WHERE Grade = 10 AND Type = BTP
```

Condition of Grade = x can be taken and Type can be y, where x and y can take all of their possible values.

- ## Vertical Fragmentation :

Student_Info 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 and teaches are tables which have derived horizontal fragmentation. These are derived from the course_offered’s horizontal fragmentation. These are helpful in queries similar to Horizontal fragmentation.

```
SELECT Faculty.First_Name FROM Faculty,teaches,Course_Offered WHERE teaches.Faulty_Id = Faculty.Faculty_Id AND Course_Offered.CourseType = “CSE”
```

And similar such queries will be executed faster and more efficiently. Since we don't have to perform a join with Course_Offered, as we know there is horizontal fragmentation on Course_Type and teaches has derived horizontal, 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_info	Vertical	1,2	2,3	3,1
Course_Offered	Horizontal	1,2	2,3	3,1
Course_Offered_Prerequisite	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
BTP_IS_Hon	Horizontal	1,2	2,3	3,1
Faculty	Vertical	1,2	2,3	3,1
Faculty_Qualification	Vertical	nil	nil	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.