CS-309
DVAAR

Abhinav Dixit (B16003) *, Vishnu Priya Jindal (B16041)†, Lokesh Kumar (B16061) ‡ and Anand Ramrakhyani (B16124)§
* b16003@students.iitmandi.ac.in
† b16041@students.iitmandi.ac.in
‡ b16061@students.iitmandi.ac.in
§ b16124@students.iitmandi.ac.in

I. Project Statement

The Ministry of Human Resource Development, Government of India, has to put an enormous effort to search for the names of all the experts working under a particular field of research, in order to invite them for a national level conference/workshop. This task often requires the involvement of all the institutions concerned as the MHRD writes to them inviting experts on the particular field. This process is generally time-consuming and tedious. This is just an instance, where any agency or some research scholar would want to identify a subject expert for some reason.

Thus, **"A centralized database of experts (faculty members, scientists, researchers), that efficiently manages and displays all the information related to the experts as well as sorts the lists of experts on various parameters, while automatically updating itself periodically"** is the need of the hour.

In our project, we focus only on the experts associated to all the Indian Institutes of Technology (IITs).

II. Requirement Analysis

A. Functional Requirements

The main operations performed under this project can be divided into two modules :

1. The Web-Crawler, which :
(a) fetches the data from the official websites of all IITs,
(b) stores the data at the relevant place,
(c) updates the database from time-to-time.

2. The Database Management System, which quite efficiently displays the results of different queries. Some of them are :
(a) searching the list of experts and sorting them on the basis of :
(i) subject of expertise / field of research,
(ii) professional experience,
(iii) academic qualifications,
(iv) associated institute,
(v) position(s) held.
(b) displaying the association between two or more experts,
(c) displaying the link to the home page of experts.

B. Non-functional Requirements

When implemented completely and publicized appropriately, we expect many queries each day and in such case the following performance parameters are anticipated :

TABLE I: Operations and their expected frequency volumes.

Serial Number	Name of Opeartion	Frequency (per day)	Required Response Time
1	Search based on subject of expertise	8k-10k requests.	Low (<3s)
2	Search based on experience	4k-5k requests.	Low (3s-4s)
3	Search based on qualifications	2k-4k requests.	Medium(3s-6s)
4	Search based on institute	6k-8k requests.	Low (2s-3s)
5	Search based on position(s) held	1k requests.	Medium (3s-6s)
6	Displaying association	500-1000 requests	Medium (<5s)
7	Displaying link to home page	1k-2k requests.	Medium(3s-6s)

III. The Conceptual Level

Based on the instances of the data fetched by our crawler program, we have set out our conceptual schema as follows :

A. Base Fact Types

F1 - Expert has name.
F2 - Expert has designation.
F3 - Expert has phone number.
F4 - Expert has address.
F5 - Expert has DVAAR Id.
F6 - Expert has e-mail address.
F7 - Expert has a profile link.
F8 - Expert works in an institute.
F9 - Expert works in department.
F10 - Expert has publications.
F11 - Expert has done sponsored projects.
F12 - Expert has professional background.
F13 - Expert has educational qualifications.
F14 - Expert specializes in some area.

B. Constraints

C1 - Each expert has exactly one name.
C2 - Each expert has exactly one designation.
C3 - Each expert has at most one phone number.
C4 - Each expert has at most one address.
C5 - Each expert has exactly one DVAAR Id.
C6 - Each expert has at most one email-address.
C7 - Each expert has exactly one profile link.
C8 - Each expert works in exactly one institute.
C9 - Each expert works in exactly one department.

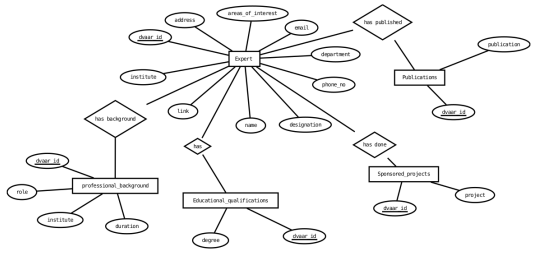

Fig. 1: Entity relationship diagram of the database.

IV. The Logical Level

The above conceptual schema is mapped to a logical schema. Since we are using the relational model, our logical schema is therefore referred to as the *relational schema*.

• Here, we intend to use 5 different tables for reflecting the 14 base facts, which were discussed in the conceptual schema. The names of these tables are unique, and are in the following order : *Expert, Publications, Professional_Background, Educational_Qualifications* and *Sponsored_Projects*.

• The constraints C1,C2,C5,C7,C8 and C9 can be implemented in the relational schema by (a) specifying the columns in the table *Expert*, and (b) by ensuring that the entries in *dvaarId* column of the *Expert* table must be unique, which in turn is enforced by declaring *dvaarId* as the primary key for this table.

• The constraints C3,C4 and C6 can be simply implemented by declaring primary keys for the tables, in which their respective entities are reflected.

• The detailed relational schema can be stated as below :

Expert (dvaarId,name, designation, phone, address, email, institute, department,link,interest_area)
Publications (dvaarId , publications)
Professional_Background (dvaarId , duration, role, institute)
Educational_Qualifications (dvaarId , qualification)
Sponsored_Projects (dvaarId , project)

V. The Physical Level

We have used the MySQL Database Management System to implement the above mentioned relational schema. The coding languages used to implement the query based search application system are Python2, PhP, HTML and CSS. The web crawler was implemented using the python library named *Beautiful Soup*.

The functional dependencies for all the tables are mentioned below :

A. Expert

1. dvaarId → name, designation, phone, address, email, institute, department, link, interest_area.
2. link → name, designation, phone, address, email, institute, department, dvaarId, interest_area.

Both the attributes *dvaarId* and *link* are candidate keys, but we select *dvaarId* as the priamary key.

One point worth mentioning here is the fact that the attribute *interest_area* is not kept multi-valued, rather it has been stored as a string. The reason for selection of this choice is that our search command was observed to perform better on this string, in order to search for all those academicians having some common interest area.

B. Publications

1. dvaarId, publications → dvaarId, publications.

Hence, the combination of *dvaarId* and *publications* is selected as the primary key.

C. Professional_Background

1. dvaarId, duration → dvaarId, duration, role, institute.

Hence, the combination of *dvaarId* and *duartion* is selected as the primary key.

D. Educational_Qualifications

1. dvaarId, publications → dvaarId, qualification.

Hence, the combination of *dvaarId* and *qualification* is selected as the primary key.

E. Sponsored_Projects

1. dvaarId, project → dvaarId, project.

Hence, the combination of *dvaarId* and *project* is selected as the primary key.

F. BCNF

All the 5 tables were observed to satisfy the following conditions :

1. None of them has any composite attributes, multivalued attributes or nested relations.

2. All non-prime attributes are fully dependent on the super key(s) of their respective tables. This is true since all super keys for tables except the *Professional_Background* have only one attribute in the super key(s), while for the *Professional_Background* table, the two attributes *role* and *institute* depend on both the prime attributes.

3. For all functional dependencies, F $\rightarrow$ G, F is always a super key. This is true for all the 5 tables.

Since the schema follows all the requirements of being in a BCNF, we conclude that our proposed relational schema, on conversion into physical schema, is in BCNF.

VI. The External Level

The UI of our application works in the following way : the opening page is the home page. It contains a search bar and a brief description about our project. In the search bar, the user can enter any keyword and the most relevant searches to the keyword are shown on the next page. The search includes keywords from name, designation, area of interest, institute name, educational background, publications etc.

After clicking on any of these profiles, the user is directed to a page where detailed profile of that professor/scholar is shown. The page also displays most relevant suggestions to a given search.

VII. Some other implemented concepts

A. Indexing

The use of indexing substantially reduces the time required for some queries. This was evident from the fact that when we search some academician/scholar by his/her name, the following two performance results were obtained in two different cases : (a) without indexing, and (b) with indexing.

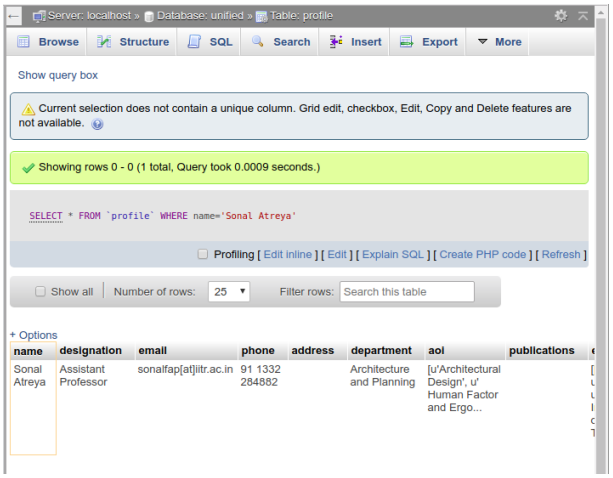

Fig. 2: Search Result in 0.0009 seconds without indexing.

index.png index.png

Fig. 3: Search Result in 0.0005 seconds with indexing

B. Query Optimization

Although, the MySQL Database Management System internally optimizes each search query very efficiently, we too have been careful enough to use the most efficient equivalent query, in first place. To achieve this , we have made use of heuristic query optimization techniques which includes using some trivial rules to write queries, which are guaranteed to give efficient queries.

Though this optimization technique does not affect the response time by any significant amount, it definitely helps one to write efficient equivalent queries, which in turn can prove to be helpful for those working in the software development field.

VIII. Our Experience of doing this Project

Building upon the practical concepts learned in the course CS-207, we formally studied about the theory of Database Management Systems and how they work on the atomic level. Thus, while developing this project we were more concerned

about implementing those concepts into modeling a real world complex problem, and subsequently transforming them into an effective database system. Thus, we had great experience in learning about the various modeling languages, the different normalization forms, etc.