**MIICT Project Report**

**(2020-2021)**

**Edgar Frank Codd**

**Submitted To**

**Prof. Ravi Prakash Gorthi**

**Team Members :**

Anupam Shah(17ucs031)

Ashish Kumar(17ucs038)

Dharmesh Poddar(17ucc022)

Lovee Hingorani(17ucs083)

Parth Patel(17ucs107)

Sanskar Bindal(17ucs141)

Shivam Saluja(17ucc075)

Ujjwal Saraswat(17ucs172)

Vatsal Agarwal(17ucs175)

Vibhor Garg(17ucs177)

Vinayak Gupta(17ucs179)

**About Edgar F. Codd**

Edgar Frank Codd born on 23rd of August 1923, to a leather manufacturer and school teacher, in Fortuneswell on the Isle of Portland in Dorset, England was famously known as the father or creator of the ‘Relational Database Model’ which is known as the foundation of Relational Database Management Systems (RDBMS). He received several honors from prestigious organizations for his work like Association of Computing Machinery, the British Computer Society, the National Academy of Engineering, the American Academy of Arts and sciences and the Institute of Electrical and Electronics Engineering also he got Turing Award in 1981 (the highest honor for computer science for his work on DBMS). He also served as a pilot in the Royal Air Force during World War II.

He attended Poole Grammar School and after graduation studied mathematics and chemistry at Exeter College in Oxford University. He also received a doctorate degree in Computer Science (1961-1965) from the University of Michigan.During 1948 he moved to New York to work for IBM as a programmer, specifically to work on one of the largest Electronic computers owned by IBM at that time. During his work there he invented a novel “multi-programming” method for IBM 7040 STRETCH computer, which allowed to run several programs at the same time. He also moved to Canada during 1953 over some personal issues but later retired to the USA and started working for IBM again. During the 1960's and 1970’s Edgar started to work on his theories of data arrangement based on mathematical set theory. He wanted to store data in tables which can be cross referenced thus allowing data permutations to be in a large amount. His seminal paper “A Relational Model of Data for Large Shared Data Banks”, published in 1970, described a new way of structuring data using ideas of set theory that eliminated the need for knowledge of internal structure of a database, opening it to people with different backgrounds to work with it. He worked with IBM for most of the time of his life and was appointed as IBM fellow in 1976.

Edgar Retired from IBM at the age of 61 in 1984 because of a fall which resulted in an injury. Later he started a company “Codd and Date Consulting Group” with his wife for conducting seminars and advising database vendors. During the early 1990’s he ceased working due to his health issue. Edgar Frank Codd died at the age of 79 on 18th of April 2003 in Williams Island, Aventura, Florida, United States Of America. The reason for his death is estimated to be a heart failure.

Theory and Practice of Database Management Systems

The term Relational Database management system was invented by E. F. Codd at IBM in 1970 and was introduced in paper A Relational Model of Data for Large Shared Data Banks However no commercially available products complies to all the relational model conform of Codd’s Rule so it is to describe broader class of relational system that is to present data to user as a relation that is in tabular form and the other is to provide relational operators to manipulate data in tabular form

First systems were implementations of the relational model were from:

1. Micro DBMS

It was the first large scale set theoretic database management system that was used in industry It permits user to define, manipulate, interrogate etc It was an interactive system It was very powerful in terms of complexity and also did included statistical computation

1. IBM UK Scientific Centre at Peterlee

It was world’s first RDBMS implemented at IBM UK implemented true model and was not having much facilities

1. PRTV

It was first model that can handle much volume data implemented relational model , can handle tables with large rows and columns also had geographic system based on RDB

RDBMS in simple is view of data as collection in tables RDBMS products implements some Codd Rules and not all

In 2009 most commercially successful employ SQL as their query language .Query languages were also implemented and proposed.

Relational Model

Relational model was invented by Edgar F. Codd. It was promoted subsequently by Chris Date and Hugh Darwen. Relational model was an approach in database management for management of data with the use of a good structure and a language that is conforming with the first order predicate logic, as described by Edgar F. Codd. All the data is represented in tuples and grouped into relations. This database organised in terms of relational model is known as relational database.A software system called RDBMS (Relational Database Management System) is used to maintain databases. Many relational database systems have an option to choose SQL (Structured Query Language) for querying and maintaining the database. These SQL databases deviate from the relational model in many ways on which Codd fiercely argued also.

The purpose of this relational model is to provide a declarative method to specify data and queries i.e. users can directly state what information the database is containing and what information the user wants from it and let database management system software handle describing of data structure for storing and retrieving the data.

In this model, the data is organised into one or more tables (also called relations) of columns and rows, each row having a unique key to identify the row. Rows are called records or tuples and columns are called attributes. Each table or relation represents one entity type (like Student or Professor). The rows represent instances of the entity type and columns represent the values associated with those instances (like Name, Age). An example could be :- each row of a course table correlates to a course, and a course correlates to a number of students, so we can say that the relationship between the course and student is - one to many.

A Relational Model of Data for large shared data banks

Future users of databank are not supposed to know about how data is stored in machines(in internal representation). A prompting service which gives these details is not best and satisfaction. User activities at terminal level and most application programs should remain unaffected whenever internal representation of data is changed and also when Some aspects of external data representation changes.

Change in data representation leads to change in data queries, update and report traffic and natural growth in the types of stored information.

All well- Known information systems which are marketed fail to make clear distinction between order of presentation on one hand and stored ordering on other. Some of the solutions were indexing dependence, access path dependence. Consider an example of a data bank which includes relations concerning parts, projects and supplies. One of relation call part is defined on following domains:

1. Part number
2. Part name
3. Part color
4. Part weight
5. Quantity on hand
6. Quantity on order

And also possibly for other domains. Each of these domains is a pool of values, some all of which may be represented in the data bank at any instant. So shown in above part example relational model of data for large shared banks has been designed.Already existing structure, formatted data systems provide users with tree-structured files or minor more general network models of the data.

## **ACID Transaction Properties**

Transaction is an atomic work where a sender sends and the receiver receives. To perform transactions in DBMS, certain properties need to be followed so that data remains consistent, accurate and integrity between the data is maintained. These properties are ACID properties-

1. Atomicity- Atomicity means either all or none. It means either the transaction needs to be performed completely or none at all. It means that transactions should not be left half or partial completed.

It uses 2 operations -

1. Abort- In aborted transactions, changes are not visible to database.
2. Commit- Once transactions are committed, changes are visible to database.
3. Consistency- Database should remain consistent before and after the transaction. It means integrity between transactions should be maintained. The total amount before transaction and after transaction must be same.
4. Isolation- It ensures that multiple transactions are possible to occur simultaneously. It means all transactions can be performed concurrently. It also ensures that changes made to one transaction will not affect the other transactions. Each transaction will be carried out as if it is the only transaction in the system.
5. Durability- Durability ensures that when transactions are complete, all changes and updates should be stored permanently so that no data is lost. Even if the system fails or restarts, no data is lost and it remains committed.

**Database Design**

Database Design is like a process which provides us the insights of the designing, implementation, development and also the maintenance of any database management system.

*Why is Database Design important?*

It helps in keeping track of all the requirements of the client and also ensures the high performance of the database.Like Software Development Life Cycle, we have Database Development Life Cycle too, where we usually have the same steps, Requirements, Designing, Implementation.

*Database Design Techniques?*

1. *ER Modelling:* Entity Relationship Model is a graphical way of representing the database design.The data model is defined on high-level considering all the data elements and their relationships for the given software. It is based on representing the real-world objects.

The word entity means a thing or object which can have a set of properties and each property can have some value.

1. *Normalization:* It is a technique which helps in reducing the data redundancy and also handles the undesirable properties like Insertion, Update or Deletion Anomalies.

It works by dividing the larger tables into smaller tables and creating the links between them using different relationships.

In SQL we achieve Normalization by simply eliminating the redundant data and ensuring that all data is stored systematically.

Different Types of Normalization are: 1NF,2NF,3NF,BCNF (Boyce-Codd Normal Form)

**Codd’s Theorem:-**

**Codd's theorem** states that relational algebra and the domain-independent relational calculus queries, two well-known foundational query languages for the relational model, are precisely equivalent in expressive power. If a query is created then it must be transformed into the other language.

This theorem accurately represents that domain-independent relational queries are invariant queries under selecting domain of values over those that appear in the database itself. That is, queries that could return different results for different domains were not included. An example of a forbidden query is the query "select all tuples except those that occur in relation R", in which R has a relationship to the database. Taking different domains, i.e., sets of atomic data objects on which subtitles can be created, this question returns different results and is therefore not independent of the domain.

Codd's Theorem is remarkable because it lays out the equality of two very different languages: relational algebra is a variable free language, while relational calculus is a logical language with variables.

Query languages equal to the ability to express relational algebra were called relationally complete by Codd. By Codd's Theorem, this includes relationship counts. The completeness of the relationship does not mean that any question of the database can be expressed in complete related languages. Well-known examples of inexpressible queries include simple aggregations (calculating tuples or summing up values that occurred from tuples, which is a clear functionality in SQL but not in relational algebra). Codd's theory also does not address the SQL nulls and the three-valued logic; the logical management of null remains the focus of the debate. In addition, SQL has multiset semantics and allows duplicate rows. However, relational completeness constitutes an important yardstick by which the expressive power of query languages can be compared.

**Database tuning**

A group of practises used to improve and homogenise the output of a database is referred to as database tuning. It typically overlaps with query tuning, but it applies to database file design, database management system (DBMS) application selection, and database environment configuration (operating system, CPU, etc.).

Database tuning helps to make the best use of system resources in order to complete tasks as quickly and efficiently as possible. While most systems are configured to control their use of system resources, there is still space for improvement by customising the database and DBMS settings and configuration.

**Input Output tuning**

RAID levels and configuration, block and stripe size distribution, and the configuration of discs, controller cards, storage cabinets, and external storage devices such as SANs are all studied in terms of hardware and software. Transaction logs and temporary spaces consume a lot of I/O and have an effect on database efficiency for all users. It's difficult to put them in the right place.

Tables and indexes that are often joined are placed such that when they are requested from file storage, they can be retrieved in parallel from different discs at the same time. To balance I/O and avoid read queuing, frequently accessed tables and indexes are put on separate discs.

**DBMS tuning**

### DBMS users and DBA experts

DBMS tuning is the method of fine-tuning the database management system (DBMS) as well as the memory and computing capacity of the machine that runs the DBMS. This is usually accomplished by configuring the database management system, but the resources required are shared with the host system.

Setting the recovery period (the time required to restore the state of data to a specific point in time), assigning parallelism (the division of work from a single query into tasks allocated to various computing resources), and network protocols used to interact with database users are all examples of DBMS tuning.

Data, execution plans, procedure cache, and work space are all allotted memory. Since accessing data in memory is much quicker than accessing data from storage, keeping a large cache of data allows operations to run more quickly. Workspace is given the same consideration. When execution plans and procedures are cached, they can be reused instead of having to be recompiled each time they are required.It's important to use as much memory as possible while having enough for other processes and the operating system to use without unnecessary memory paging to storage.

To increase concurrency, processing resources are often delegated to unique activities. On a server with eight processors, six could be set aside for the database management system to optimise the database's computing power.

### Automatic DB tuning

Machine learning is used to learn how to assess output under a variety of workloads.

**Database maintenance**

Backups, column statistics updates, and defragmentation of data inside database files are also part of database maintenance.

The transaction log of a heavily used database grows quickly. To make space for future entries, transaction log entries must be deleted. Since frequent transaction log backups are smaller, they cause less downtime in the database.

To find data in a set against a table or index, DBMS use statistical histograms. Statistics updates should be performed on a regular basis, with as much of the underlying data as possible being sampled. In order for query engines to make good decisions about execution plans and efficiently locate data, they need accurate and up-to-date statistics.

Table and index data defragmentation improves data access performance. The degree of fragmentation is determined by the quality of the data, how it is updated over time, and the amount of free space in database pages available to accept data inserts without creating new pages.

**Future Databases Modifications:**

After Edger Frank Codd laid the base for the Relational Database Management System which was used to invent DBMS like SEQUEL (which was later renamed to SQL) and was popular for most of the time. Many others Relational Database Management System were also discovered using these as basics but later in the early 21st century many databases management system were invented which does not uses Codd’s theorems and became pretty popular for their features like MongoDB, Cassandra which were also named as NoSQL databases and by time they have only gained popularity on the expense of database management systems like MySQL.

**References:**

1. <https://www.britannica.com/biography/Edgar-Frank-Codd>
2. <https://history-computer.com/edgar-codd-creator-of-the-relational-database/>
3. <https://en.wikipedia.org/wiki/Edgar_F._Codd>
4. <https://www.techrepublic.com/article/nosql-databases-eat-into-the-relational-database-market/>
5. <https://en.wikipedia.org/wiki/Relational_model>
6. <https://en.wikipedia.org/wiki/Codd%27s_theorem>
7. <https://en.wikipedia.org/wiki/Database_theory>
8. <https://en.wikipedia.org/wiki/MICRO_Relational_Database_Management_System>
9. <https://en.wikipedia.org/wiki/IBM_IS1>