An Overview of Relational Databases

By Jason Figueroa, ECE '20

Introduction

The concept of a database is rather simple: it refers to any organized collection of data. Databases have become a popular alternative to the file-based approach to information storage because it allows for a single location for all data, and it can be shared by multiple users without the need for file duplication. This data is organized and accessed instead using a database management system, or DBMS for short. Relational databases use a DBMS known as the relational model to manage and store their data. This model relies on the use of tables to store the data, and keys to access the data. This model also uses relational algebra to describe operations on the data, and a query language known as SQL to perform these operations. Relational databases allow for organized storage, connection, and query of the data it is storing.

The Relational Model

The relational model for database uses tables, also known as relations, to represent the data and to show the relationships within this data. Each row in a table is a single item, also known as a tuple, and each column represents a different attribute of this tuple. Neither the rows nor the columns must be in any order, but every row must be unique and every entry in a column must be of the same data type. Finally, each table must have an associated key, which can be any set of attributes within the table (Sumathi 68).

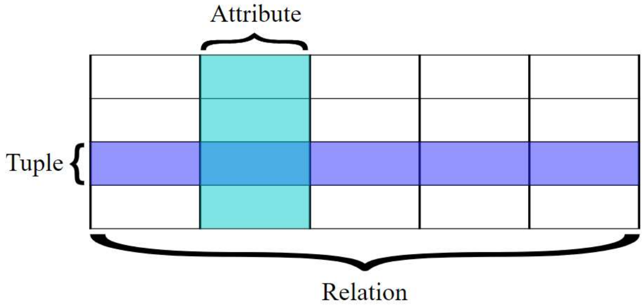

Figure 1. table example for a relational database

How Key values are used

The concept of keys is very powerful in relational database systems, for it allows for both7 distinction of tuples within a relation, and the formation of links between different relations. Any attribute or set of attributes that is capable of uniquely identifying each tuple within a relation is known as a superkey. Because superkeys generally contain more information than is necessary to properly identify a tuple, there is the concept of a candidate key, which is a superkey that has no subset of its attributes that is also a superkey. If a key consists of more than one attribute, it is known as a composite key. One of these candidate keys will used to identify tuples within this relation, and that is known as the primary key, the candidate keys not selected are alternate keys. Finally, if we want to relate two tables to each other, we need some set of attributes to match a candidate key of another relation. This is known as a foreign key. Another critical concept in relational databases describes a set of constraints the information within the database must follow to maintain accuracy of data.

Constraints

For a database to ensure that the data it is representing is accurate, it follows a set of integrity constraints. In relational databases, there are two primary types of integrity: entity integrity and referential integrity. Entity integrity ensures that no attribute of the primary key can be null, or undefined, for that tuple. If this integrity were to be broken, this then implies that this attribute is not needed to distinguish between tuples, and therefore the key we are using is not a candidate key. The second type of integrity, referential integrity, states that if a foreign key exists in a relation, it must either match a candidate key value of some tuple in its home relation, or the foreign key must be null. For example, if the studentID attribute in the Course relation is a foreign key targeting the Student relation, then a studentID that does not exist within the Student relation cannot be added to the Course relation, however we can create a new tuple in the course relation and set the studentID attribute to null. Other than these two constraints, additional rules can be set by the user, and these are known as general constraints (Connoly 82-83). Aside from these constraints, all operations on a relational database are described in their own language, known as relational algebra.

Database Operations

Databases are a complex data structure in computer science, and therefore there must be some schema to describe how we retrieve the information from the database. For relational databases, there are two systems that describe how we can retrieve the information we want: relational algebra and relational calculus

Relational Algebra

Relational Algebra is a theoretical language that describes the operations that can be performed on one or more relations. These operations do not change the input relations, and the outputs are relations themselves. This is a property of relations known as closure, where the output of one relational algebra operation can be the input to another. There are two types of relational algebra operations: unary, which work on only one relation, and binary, which work on pairs of relations. There are only two unary operations, selection and projection, while all the other operations are binary.

Relational Calculus

While relational algebra describes how to retrieve desired data from a set of relations, there is a very similar model that instead describes what we want to retrieve from the relations. This is known as relational calculus, and it is based on predicates, which is defined as a truth-valued function with arguments, and propositions, which is when we substitute values for arguments in a predicate. There are two main forms of relational calculus: tuple relational calculus and domain relational calculus. Tuple relational calculus is used to find tuples in which a given predicate is true. Tuple relational calculus relies on the use of tuple variables, which is a variable whose values can only be the tuples of the desired relation. Domain relational calculus, on the other hand, the variables used get their value from domains of attributes rather than tuples within the relations. While relational algebra and relational calculus are different, it is possible to rewrite any expression from relational algebra to relational calculus, and vice versa (Connoly 103-107). The standard management system for relational databases uses relational calculus as to help form its instructions

SQL

The Structured Query Language, better known as SQL, has become the standard DBMS for relational databases. This is a language where the user inputs what we require, rather than how to get it. SQL is used to make calls to the database and perform tasks, such as add or read data, and it has six main instructions: SELECT, INSERT, UPDATE, DELETE, CREATE, and DROP. The select command is the most frequently used and the most important query in SQL, and it can retrieve and displaying data from one or more tables, and it can complete the selection, projection, and join operations from relational algebra. The select instruction also has many clauses, which modify the output of this instruction. An example of this would be the WHERE clause, which selects out rows to be included in the select based on the condition given. The other five instructions are much simpler, and complete the task implied by their names. SQL is a very powerful DBMS and has become the industry standard for relational.

Conclusion

The creation of databases allowed for more efficient data storage, and the creation of relational databases allow for similar data to be stored together in a table. Relational algebra and relational calculus describe how operations to this table must be done, and SQL provides the syntax for us to use these operations. The use of keys allows the user to access any tuple within the database using a minimum number of items and allows users to relate different tables to one another. Relational databases are a powerful tool in modern computing and will be an integral part of future projects.

References

1. Connolly, T. M., & Beg, C. E. (2015). Database systems: a practical approach to design, implementation, and management. Boston: Pearson. http://vumultan.com/Books/CS713-Database-System-A-Practical-Approach-to-Design-Implementation-and-Management-by-T-Connoly-C-Begg.pdf

2. Sumathi, S., & Esakkirajan, S. (2010). Fundamentals of relational database management systems. Berlin: Springer. http://vumultan.com/Books/CS713-Database-System-A-Practical-Approach-to-Design-Implementation-and-Management-by-T-Connoly-C-Begg.pdf

3. Codd, E.F. “A Relational Model of Data for Large Shared Data Banks .” Communications of the ACM, vol. 13, no. 6, June 1970, pp. 377–387 https://www.seas.upenn.edu/~zives/03f/cis550/codd.pdf