SUPCOM – MODULE 3

Database Methodology

Gabriel Castagnino

TASK DB1

1) Data is a key element to the base of information and knowledge of any kind of organization. It seems intrinsically consequent that investing in a database system is going to foster then the organization's capability to manage (organize, create, delete, etc) and use all the data in its own favor, in an efficient way. The better the data moves in every way planned and desired, the better the information the organization has, the better the profit it can take from it.

Apart from that main and broad principle, a database system introduces a simplified format for all the data available for an organization. All data is then integrated into the same system, and shared by all users, which is even better with the possibilities offered by subschemas. In this sense, investing in a database system will not only allow us to share all information but also share as needed.

2) A database is gathered data in tables with relations between their rows, and the metadata that explains its structure, and it is often stored on disks. A DBMS is basically a software program that creates, process and manages databases. A database system combines these two so as to convert large data conglomerates into accessible, manageable and useful sources of information.

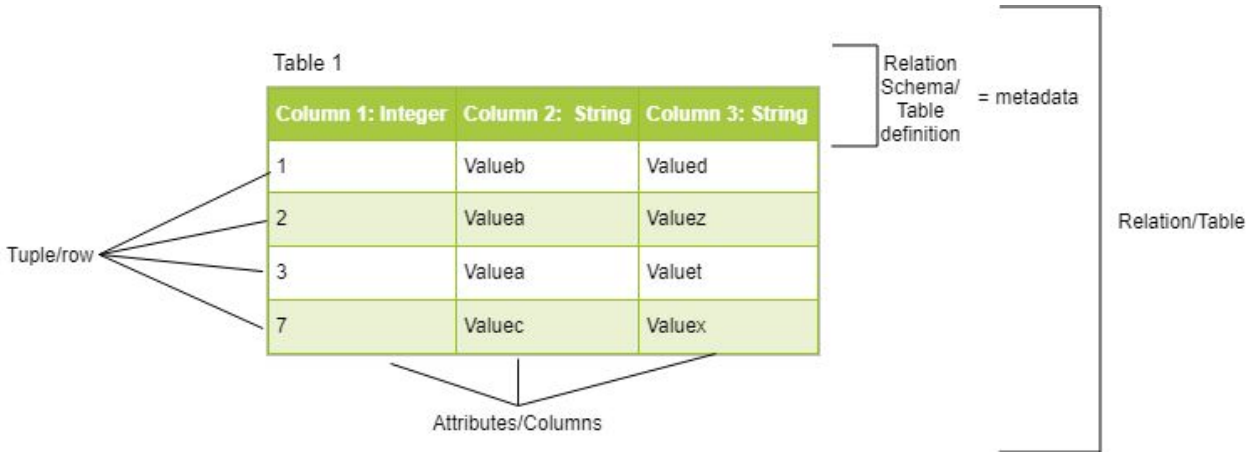
3) A three-tier has a more efficient network and processing thanks to the extra layer. The two-tier still has all users sending requests directly to the server. Once there are more users or request than its capability, the network becomes filled, or the server ultimately not responding. Also, a three-tier is more secure, since the clients do not have direct access to the database, but to the APPS instead.

4) A three-layer Architecture can be put into practice for example in online sales systems like Amazon. Users (customers) can access an enormous amount of data of their interest -products- that are mainly only a subschema of the database of Amazon. The user interface they access is connected through the internet to an APPS that solves their requests communicating safely with the DBMS, and this of course with the DB.

TASK DB2

5) A tuple is an ordered list of one or more values or properties. It can be considered as an entity that has attributes, and these attributes are characteristics. A relation is basically a set of tuples, in which each tuple is unique and the tuple order is irrelevant. The relation's structure is characterized in a relation schema; it is metadata about the relation. Metadata is a description to aid in the understanding of the data. A relational database schema is composed of many relations schemas that are logically connected. Therefore, the relational database schema is the metadata about those relations and connections.

In the relational model, relations are implemented as tables. The data stored becomes quite easy to understand visually as tables logically connected to each other. In this sense, the relations schemas become the tables' definitions, the attributes become columns and the tuples rows.

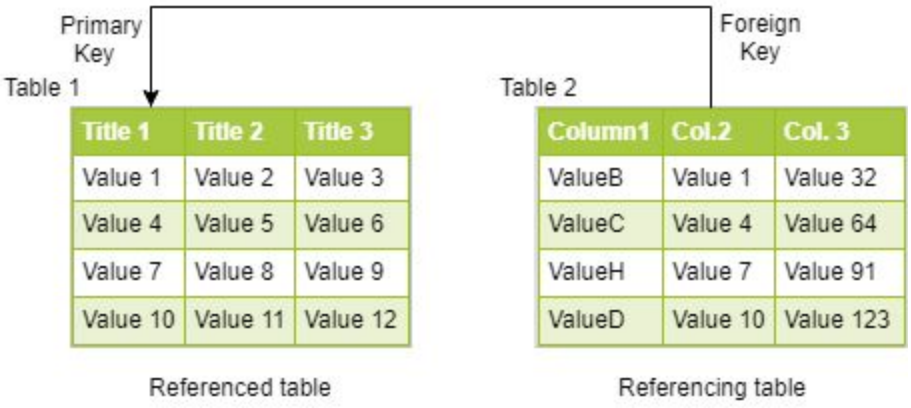


6) NULLLs are not a value and therefore they cannot be compared to anything. Often they could be expressing a bad database design, or a value not available or unknown to the database. Since relational databases work on the basis of relations, a NULL value makes a tuple attribute impossible to be compared to. In this sense, when choosing a PK we must have entity integrity, for later constructing referential integrity.

7) A Candidate Key (CK) is, in other words, the minimal Super Key. This means that the CK is composed of all the unique attributes or columns that identify a unique row in a table/relation. Every

table must have at least one or more CKs. From this point, further on there is the Primary Key (PK), which is the one chosen by the designer of the database among the CKs to identify that relation. And there must always be only one PK per table (composite or not). In contrast, Alternate Key (AK) are all the other CK that have not been chosen for this purpose. And finally, a Surrogate Key (SK) is like an artificial CK that contains no real information, but that it is created as an integer for finding tuples and referencing relations, generally used due to PKs related problems (composite hard handling, uniqueness over time, disagreement between users).

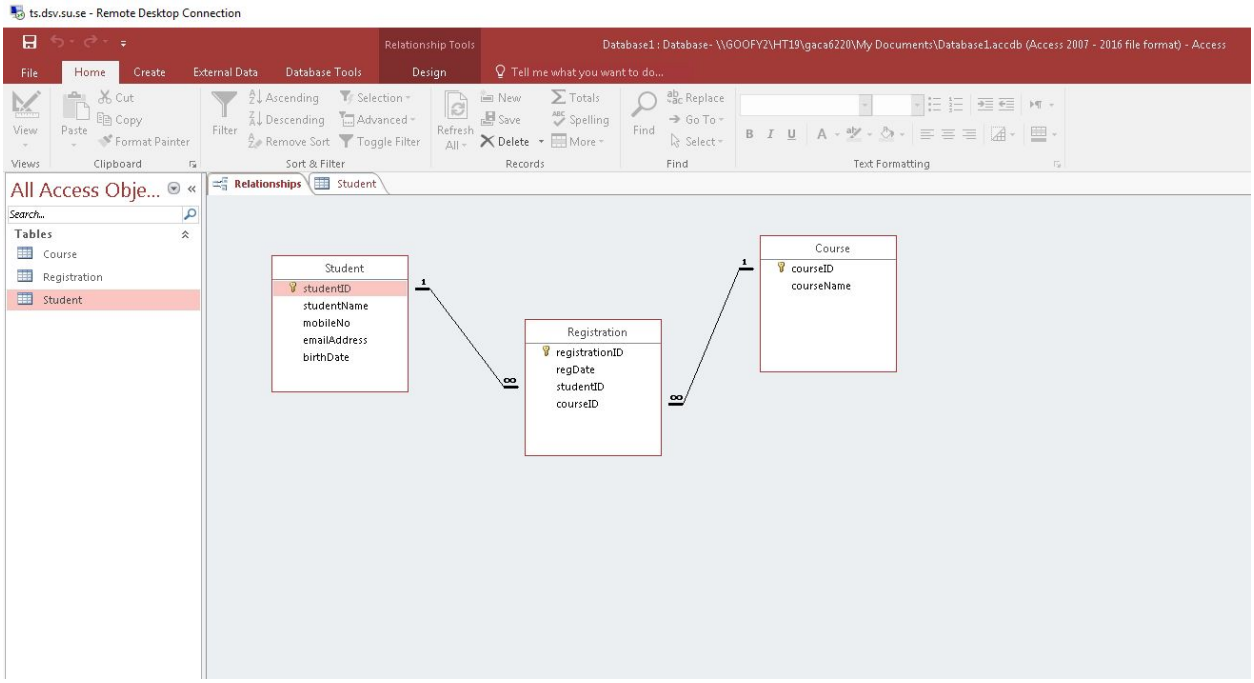
8) FKs connect the referencing table with the referenced table, via duplicating the PK of the referenced table as an FK in the referencing one. Consequently, the FK becomes the connecting reference to a relation. They represent connections in a relational database. And in order to create referential integrity, all the attributes of the FK must match the attributes of the PK referenced.



Title 1	Title 2	Title 3
Value 1	Value 2	Value 3
Value 4	Value 5	Value 6
Value 7	Value 8	Value 9
Value 10	Value 11	Value 12

Column1	Col.2	Col. 3
ValueB	Value 1	Value 32
ValueC	Value 4	Value 64
ValueH	Value 7	Value 91
ValueD	Value 10	Value 123

TASK DB3


ts.dsv.su.se - Remote Desktop Connection
Relationship Tools
Database1 : Database- \\GOOFY2\HT19\gaca6220\My Documents\Database1.accdb (Access 2007 - 2016 file format) - Access
File
Home
Create
External Data
Database Tools
Design
Tell me what you want to do...
All Access Obje...
Search...
Tables
Course
Registration
Student
Relationships
Student
Student
studentID
studentName
mobileNo
emailAddress
birthDate
Registration
registrationID
regDate
studentID
courseID
Course
courseID
courseName
1
∞
1
∞

TASK DB4

10) Relational database schema in the form of textual description

Person (personID, name, address, libraryCardNo)
Librarian (personID, employeeNo)
Borrower (personID)
BookLoan (bookLoanID, loanDate, personID, bookCopyID)
BookCopy (bookCopyID, copyNo, bookTitleID)
Purchase (purchaseID, purchaseDate, bookCopyID)
BookTitle (bookTitleID, title, ISBN)
BookAuthor (bookAuthorID, bookTitleID, authorID)
Author (authorID, authorName)
BookCathegorization (bookCathegorizationID, bookCathegoryID, bookTitleID)
BookCathegory (bookCathegoryID, cathegoryName)

FKs

Borrower.personID is FK to Person.personID
Librarian.personID is FK to Person.personID
BookLoan.personID is FK to Borrower.personID and to Librarian.personID
BookLoan.bookCopyID is FK to BookCopy.bookCopyID
Purchase.bookCopyID is FK to BookCopy.bookCopyID
BookCopy.bookTitleID is FK to BookTitle.bookTitleID
BookAuthor.bookTitleID is FK to BookTitle.bookTitleID
BookAuthor.authorID is FK to Author.authorID
BookCathegorization.bookCathegoryID is FK to BookCathegory.bookCathegoryID
BookCathegorization.bookTitleID is FK to BookTitle.bookTitleID

AKs

Librarian.EmployeeNo AK
BookLoan.(loanDate, personID, bookCopyID) AK1

BookCopy.(copyNo, bookTitleID) AK1

Purchase.(purchaseDate, bookcopyID) AK1

BookTitle.ISBN AK

BookAuthor.(bookTitleID, authorID) AK1

BookCathegorization.(bookCathegoryID, bookTitleID) AK1

BookCathegory.cathegoryName AK

TASK DB5

11) Normalization comprehends a series of formal steps, or ‘normal forms’, mainly focused on “converting poorly structured tables into two or more well-structured tables” (Kroenke & Boyle, 2017, p. 472). It is useful to save storage space, since every data item must be in only in one table. But even better, it improves the data quality due to the fact that it prevents issues when updating, creating or deleting data items in the database.

12) In RDB design, the data model provides a “great picture, and the normalization contributes with the finer details” (Church, 2012, p. 123). It means it is a complementary way for designing the best possible structure of the RDB. In this detail-level, normalization ensures that the values in columns are flattened, attributes are in the right tables, and, if necessary, more tables are added to keep our data accurate.

13) In contrast to what can be called a universal relation, or a single spreadsheet containing all the information we want, a normalized database implies having many tables. So many as needed for the data structure to be finally normalized. A drawback is that evidently a RDBS will need to trace a single unique data item scavenging through keys until it reaches the appropriate row, and this slows up the processing.

14) Functional dependencies occur when a column (or several) in a table seem to hold a dependency on another(s). This means that when there is a functional dependency, if you know a value for column A, you should also know a unique value for column B. It also could be represented A —> B, meaning that A determines B, and A is, of course, the determinant. It plays a central role for normalization, since its ultimate goal is to have only the primary key as the determinant of all other columns in each table.

15) A surrogate key (SK) is a unique column that it is created so as to play the role of the primary key. With the SK we should know all the other values of the table if it is properly normalized. It helps to solve the 2NF since there will not be a composite primary key. However, there could still be transitive dependencies on the table, meaning columns that can uniquely determine other values as well in the table. It is in the 3NF, that this situation becomes normalized, because any such other column is open as a new table.

TASK DB6

16) What is the ISBN for the book title The Little Prince?

BookTitle

bookTitleID	title	ISBN
1	The Institute	9781982110567
2	Republik	9780465094080
3	The Little Prince	9780749707231
4	The Prince	9780486272740

SELECT isbn

FROM BookTitle

WHERE title = 'The Little Prince'

ISBN
9780749707231

17) What is the author of each book?

BookTitle

bookTitleID	title	ISBN
1	The Institute	9781982110567
2	Republik	9780465094080
3	The Little Prince	9780749707231
4	The Prince	9780486272740

BookAuthor

bookAuthorID	bookTitleID	authorID
1	1	1
2	2	2
3	3	3
4	4	4

Author

authorID	authorName	
1	Stephen King	
2	Plato	
3	Saint-Exupéry	
4	Machiavelli	

```
SELECT title AS book
       authorName AS author

FROM BookTitle BT, BookAuthor BA, Author A

WHERE BA.bookTitleID = BT.bookTitleID
AND BA.authorID = A.authorID
```

book	author
The Institute	Stephen King
Republik	Plato
The Litlle Prince	Saint-Exupéry
The Prince	Machiavelli

TASK DB8

19) When administering a database, especially in the process of updating a large database, there could be several problems related to the heavy load of transactions. Sometimes, these transactions or operations are partially completed due to several factors, and it could lead to inaccurate information. That is the reason why a DBMS has features that deal with such situations, in order to keep the database integrity. And the Commit/Rollback protocol serves exactly this purpose

Prior to altering a database, such alteration is recorded in the log with all the steps or actions of it, until it reaches a commit point. This point is to where the DBMS commits itself to alter the database, should all the actions be performed correctly. But, in case a malfunction may occur, or a transaction cannot be finished, the DBMS will make use of the log to roll back or undo the actions taken. Therefore, it prevents the database from being left in an inconsistent state. Moreover, if in between other transactions have used wrong database entries due to this temporary inconsistency being corrected, the protocol resorts to a cascading rollback that will undo these transactions as well.

20) The Locking protocol of a DBMS comes in handy to prevent issues related to the numerous transactions occurring at the same time. For example, the incorrect summary problem or the lost update problem. What happens then when two or more transactions try to update values in a database base on the same columns, but with different intentions? A database needs a certain order on the transactions, and that order is build up from locks. What a DBMS does is to lock certain items that currently in use by a transaction. If that transaction is not going to modify the data, the DBMS puts a shared lock, so other transactions can still view the data. But, if it is going to alter it, the DBMS puts an exclusive lock instead, making that transaction the only one possibly affecting it during the locking time.

Moreover, the locking protocol works with the wound-wait protocol to prevent possible deadlocks. This means that the DBMS gives priority to older transactions. The new ones will have to release the exclusive access to the data and even rollback, if it is required by an older one, and then they will grow older until it is their turn.

21) A SQL stored procedure is a sequence of SQL commands and logic that is stored on the SQL server. In this sense, a user can then call that procedure to be executed whenever desired. For example, when a VIEW is created, we have only stored a query to read data from tables, using mainly the SELECT command. Therefore a VIEW could be created within a stored procedure. However, a stored procedure allows us to read but also alter data. It can insert, update or delete data, depending on the stored SQL commands. And it is not necessary that any product is returned to the user.

A trigger procedure also is a sequence of SQL commands and logic stored on the SQL server, but it is supposed to work automatically whenever a certain event happens and is recognized but such procedure. A SQL stored procedure can be called as many times as wanted by the user, however, a trigger procedure can never be called by the user. It is more an automatic response, encoded to react to events. For example, if we want to avoid deleting tables, a trigger procedure could print information about it, and rollback the action.

References

Brookshear J.G.(2012). *Computer Science - an overview* (11th ed.). Pennsylvania. Addison-Wesley.

Churcher C. (2012). *Beginning Database Design* (1st ed.). Berkeley, California. Apress..

Kroenke, D. M. & Boyle, R.J (2017). *Experiencing MIS* (7th ed.). Pearson

Kulamani S. (2013). Difference between a Stored Procedure and a Trigger. Doi: https://www.codeproject.com/Tips/624566/Differences-between-a-Stored-Procedure-and-a-Trigg

Kumar, R. (2008). View & Stored Procedure in SQL Server. Doi:https://www.c-sharpcorner.com/uploadfile/raj1979/views-stored-procedure-in-sql-server-2005/

MICROSOFT (2018). Documentación de SQL. Doi: https://docs.microsoft.com/es-es/sql/t-sql/lesson-1-creating-database-objects?redirectedfrom=MSDN&view=sql-server-2017

Pankaj P. (2013). Difference between stored procedures and triggers. doi:https://www.c-sharpcorner.com/blogs/differences-between-stored-procedures-and-triggers1