## MODULE 02

**NOTES ON CONCEPTUAL DESIGN**

**ENTITIES AND ATTRIBUTES**

**WHAT ARE WE TRYING TO DESIGN?**



A lot of this you know having done SQL first. So, will be fast doing this first part.

**A DATABASE IS A WAY OF STORING DATA.**

**A file system is another way of storing data.**

The data is stored in tables that look like 2-dimensional spreadsheets to the user (remember ACCESS tables). In fact, these tables look like files with data in them -- and just the tables can be thought of as files of data.

### What is in the tables?

It is **information about things**. How is that for an explanation? We store student information in a table and information about banking in another table. So tables contain information about things the business needs to keep track of. These things are called entities.

**WHAT IS AN ENTITY?**

An **entity is an object or event** that is of particular interest to an organization and for which **data needs to be stored**.

# TANGIBLE ENTITY

Some entities are tangible. That means they are physical and can be touched. Students can recognize these entities easily. Some examples of tangible entities are

STUDENTS,

INVENTORY,

LIBRARY BOOKS,

PRODUCTS

# INTANGIBLE ENTITY

Other entities exist because of an event that occurs within the operation of a business or expected events. You cannot touch these intangible entities, but the organization cannot exist without their existence and data must be stored for them. Some examples of intangible entities are

ORDERS,

INVOICES,

COURSES,

PATIENT TREATMENT

Once the database designer has understood the business organization, its short and long term goals, its strategic plan, he/she is in a very good position to identify the entities that must exist for the organization to carry on business, both now and in the future.

The system's courses should give you, the student, a good background in how to understand the business organization. Those preliminary activities to the entity design will not be covered on this course. For the first go through we will use a simple form or report to study the system and work toward a completed design. This means that the preliminary activities required to understand the business, discovered that this business or system produces one form or report only. There are additional cases for you to practice your skills.

The **DESIGN PROCESS** then is to decide what entities are needed and what information or characteristics of the entities we want to keep for the business to function.

The IMPLEMENTATION PROCESS is to take the design and physically build it with the software like how it was done in Oracle

# We will learn the design methodology through solving problems.

The next page shows a system. We must determine what information needs to be stored and how the information will be stored and how each piece of information relates to other pieces.

**STUDENT COURSE APPLICATION SYSTEM**

The entire system consists of only one report, the STUDENT TRANSCRIPT. Two samples of the report are shown below.

SENECA COLLEGE DATE: 2011 12 31

CAMPUS: Don Mills STUDENT NO. 1234567

STUDENT NAME: G.P.A. 2.7

Mary Kim

123 This Street (416) 925-2861

No City, ON

A9A 9A9

COURSE DESCRIPTION GRADE FACULTY NAME OFFICE PHONE

PAS145 PASCAL 2.0 CS245 TOM JONES 135 7229

COM223 COMMUNICATIONS 3.6 AP142 SALLY ALI 139 7201

MAT111 MATHEMATICS 2.4 CS153 KRIS GOLLY 130 7255

SENECA COLLEGE DATE: 2011 12 31

CAMPUS: Don Mills STUDENT NO. 7654321

STUDENT NAME: G.P.A. 2.6

John Doe

345 That Street (905) 491-5050

My City, ON

X1X 1X1

COURSE DESCRIPTION GRADE FACULTY NAME OFFICE PHONE

PAS145 PASCAL 2.0 CS245 TOM JONES 135 7229

COM223 COMMUNICATIONS 2.7 AP142 SALLY ALI 139 7201

MAT111 MATHEMATICS 3.0 CS153 KRIS GOLLY 130 7255

IMPORTANT NOTE:

The system is restricted to producing this one form or report. This was done in order to keep the process needed to develop a database as simple as possible for the first go through.

**STEPS IN DESIGN OF A DATABASE SYSTEM**

These notes will attempt to give you a process for designing databases. As you gain experience steps 1 and 2 will often be done at the same time. For now, we will try to do this in the order suggested. At the end of the processes there will be a lot more entities than initially discovered. The more experience you gain with design the more you will see these entities sooner and the result will be less work in some of the steps.

(There are other approaches to design and as long as they work for you they are all good approaches)

## STEP 1 -- IDENTIFY ATTRIBUTES

# WHAT IS AN ATTRIBUTE?

An attribute is a characteristic of the entity. We have traditionally thought of these as fields within a record. However, an attribute can extend beyond fields. A characteristic that can be represented with character or numeric data can be stored within a computer. Characteristics, such as a student’s willingness to learn, for example, can only be approximated and is difficult to record and store.

For now, go through the transcript, which represents the data that the organization wants to store. Write down all the data items you see. **Take your time as the designer** and look for all the things that the business would like to maintain information on. It is hard to design well if you do not have ALL the information. Again, please make sure you have not missed anything. (As a suggestion, using the Student Transcript, number each attribute you find and list them as follows).

Try to not look at the list on the next page until you have tried it yourself.

Let us list them on next page

SENECA COLLEGE DATE: 2011 12 31



CAMPUS: Don Mills STUDENT NO. 1234567



STUDENT NAME: G.P.A. 2.7



Mary Kim



123 This Street (416) 925-2861



No City, ON



A9A 9A9



COURSE DESCRIPTION GRADE FACULTY NAME OFFICE PHONE



PAS145 PASCAL 2.0 CS245 TOM JONES 135 7229



COM223 COMMUNICATIONS 3.6 AP142 SALLY ALI 139 7201

MAT111 MATHEMATICS 2.4 CS153 KRIS GOLLY 130 7255

SENECA COLLEGE DATE: 2011 12 31

CAMPUS: Don Mills STUDENT NO. 7654321

STUDENT NAME: G.P.A. 2.6

John Doe

345 That Street (905) 491-5050

My City, ON

X1X 1X1

COURSE DESCRIPTION GRADE FACULTY NAME OFFICE PHONE

PAS145 PASCAL 2.0 CS245 TOM JONES 135 7229

COM223 COMMUNICATIONS 2.7 AP142 SALLY ALI 139 7201

MAT111 MATHEMATICS 3.0 CS153 KRIS GOLLY 130 7255

Mark up user document, number each attribute.

Then List attributes here --

1 COLLEGE NAME

2

List of attributes or “things” that need to be stored

### Attribute name other notes

1 COLLEGE NAME Only one college and campus – Management said they use pre-printed forms



2 DATE If this is the run date of the transcript it does not get stored



3 CAMPUS Same as 1 above it does not get stored



4 STUDENT NUMBER

5 GPA This is a calculated field and is not stored



6 NAME

7 STREET For simplicity in this design 7 to 10 will be called address



8 CITY

9 PROV Address



10 POSTAL CODE

11 PHONE In this case we will allow more than one phone number to be stored



12 COURSE CODE

13 COURSE DESCRIPTION

14 GRADE

15 FACULTY NUMBER

16 FACULTY NAME

17 FACULTY OFFICE

18 FACULTY PHONE

**STEP 2 -- IDENTIFY ENTITIES**

# IDENTIFY ANY ENTITITES YOU SEE

### ENTITIES ARE THINGS THAT WE WANT TO STORE DATA ON

Each of us can look at the transcript report and see several entities. Some of us will look at this report and may see just one entity. Because we will have a very structured method of working with the entities that you identify and turning them into a database design, any entities we miss discovering at the conceptual design phase, will be identified as time goes on mostly through a process of normalization (more on this later)

# KEY -- YOU MUST FIND AT LEAST ONE ENTITY

-- EXPERIENCE WILL HELP YOU FIND MORE

If you had seen more entities the design will be easier or faster to do but let us assume that we all see two entities for this system. The first is obvious to most of us, that being the student. Also, it is apparent that we will want to keep data for each faculty member. Therefore, we can list two entities for our STUDENT COURSE APPLICATION SYSTEM now.

# STUDENT

**FACULTY**

Some people will choose as an entity ADDRESS or GRADE. Perhaps a simple way of noticing the difference is to ask the question, what characteristics of a GRADE, does the organization want to store. That would be tough to answer when compared to asking, what characteristics of a student, does the organization want to store (phone, address, age, id...)

Many of you will see another entity such as COURSES. As you gain experience you will see still other entities and that will make designing a system a little easier.

**For now, we will assume that only two entities were apparent**.

These two entities were chosen because (a) we can demonstrate that the process being applied will bring out other entities and (b) these two entities will have inherent problems that will allow us to show the entire normalization process. If we saw more than two entities there would be less to show you of the normalization process.

ASIDE: Other approaches to design like to look at single user views. The transcript is a user view or the output of data that the user can see. All the attributes are then put into a single entity in un-normalized form. The short form is UNF and it appears as follows

UNF [COLLEGE NAME, DATE, CAMPUS, STUDENT NUMBER, GPA, NAME, ADDRESS, PHONE, COURSE CODE, COURSE DESCRIPTION, GRADE, FACULTY NUMBER, FACULTY NAME, FACULTY OFFICE, FACULTY PHONE]

More on this methodology can be found in the design labs and note on the course site.

Isn’t this the same? List all the attributes. Except it has not defined any entities yet.

STEP 3 -- LIST ATTRIBUTES WITH THE ENTITIES

# LIST ALL ATTRIBUTES WITH THE ENTITY

Review each of the attributes individually and assign the name of the attribute to **only one** of the entities that it best belongs with. Remember to put it with only one entity. Make sure all attributes have been assigned to an entity or that there is a reason for not assigning an attribute to an entity (see below). **Do not miss any**.

Note: For attributes that are pre-printed on the form or run dates of reports that can come from the system at run time, or for calculated fields do not assign them to an entity. It was important to record them as required data when you implement the system, but attributes such as calculated attributes are handled through programming at run time.

Aside: Do not spend a long time trying to decide which of two entities an attribute belongs to, as it will resolve itself by following the rules and process shown later. Just put it into an entity that makes sense.

## To do step 3 requires that we cover a common notation.

## Step 3 will be shown after the explanation on notation.NOTATION USED

Notations attempt to communicate meaning through a standardized format. The notation must be agreed upon by all parties for facilitating communications.

There are many acceptable notations for listing entities and their attributes and one is not necessarily more correct than another. What is important is that there is agreement on the meaning of the notation used. One accepted and **simple** method of notation for listing attributes is as follows:

CUSTOMER [CUST-ID, CUST-NAME, STREET, CITY, PROV, POSTAL-CODE, CREDIT-LIMIT, delivery-instructions]

**The entity name is written in upper case 🡪** CUSTOMER.

**The attributes for the entity are surrounded by square brackets** [ ].

**The required attributes are listed in upper case**. That means that when data is stored in the entity about a customer there must be a value entered for every attribute. To store information about a customer the attribute CUST-ID must have a value and cannot be left blank. The same applies for name, address, and credit limit. In this case the organization must assign a credit limit to a customer BEFORE it can store any information about the customer in the table.

**Optional attributes are listed in lower case**. The example shows delivery-instructions in lower case. When information about a customer is stored there may not be any special delivery instructions such as “deliver to rear entrance”. This attribute can contain nothing.

**Attributes that repeat are surrounded with round brackets** ( ). Referring to the STUDENT TRANSCRIPT SYSTEM, in the case of students, the attribute phone would be a repeating attribute if the organization wanted to record multiple phone numbers for the student. The same would be true if the organization wanted to record multiple addresses such as current address and permanent address, for those who would normally live out of town or province. The same would apply to courses taken or enrolled in. If the student takes more than one course then the attributes course code, description, and grade repeat as a group (remember back to the first week example when you were required to make a data structure.)

**When attributes repeat as a group, the entire group is surrounded with round brackets** ( ).

The notation described above will be used throughout these modules. A look into other texts will show other notations. Students are welcome to use other notations from their systems course if the chosen notation is readily understandable.

NOTE: You would only see the brackets during development of a solution. When you get to something called 1st normal form you will see why the brackets are gone.

**STEPS SO FAR**

# After you understand the system or organization

1 Find and list **all** the attributes

2 Find **at least** 1 entity

3 Assign each attribute to one of the entities

ASSIGNMENT (10 minutes) Do step 3 above.

STUDENT [

FACULTY [

**ENTITY LISTING -- early stage**

An entity list for the STUDENT TRANSCRIPT SYSTEM based on two entities may look like:

STUDENT [STUDNO, NAME, STREET, CITY, PROV, POSTAL, PHONE,

CRSCODE, CRSDESC, GRADE]

FACULTY [FACNO, FNAME, OFFICE, FPHONE]

NOTE: Again, make certain all attributes are accounted for.

At this stage I am re-writing some of the attributes under the attribute name ADDRESS for simplicity in this example: You really do need all of them.

STUDENT [STUDNO, NAME, ADDRESS, PHONE, CRSCODE, CRSDESC, GRADE]

FACULTY [FACNO, FNAME, OFFICE, FPHONE]

NOTE: Some texts and some instructors like all the attributes to be listed in a single entity. If you as the student would prefer to do that method, please do so. Later you can change to whatever variation suits your thinking style. I found in larger systems there were just far too many attributes to put into one entity. Also, experience will have you finding the obvious entities quickly.

**NOTATION - MANDATORY OPTIONAL**

At this stage we are not required to determine what a mandatory/required attribute is versus an optional attribute. We will look at it now so we can show the notation being used and sometimes it may help in pointing out another entity.

**WHAT ATTRIBUTES ARE REQUIRED**

**WHAT ATTRIBUTES ARE OPTIONAL**

Ask the above questions to every attribute. For example, when originally storing student data ask is the attribute is required or is optional. Most personal information on students will be mandatory or required. The student would know their name and the address for mailing the transcript. If this system required storing gender, date of birth, SIN#, and OHIP# the student would know that information and it might be reasonable to expect them to be mandatory.

**The decisions made in this design process must reflect the way the business operates, meaning the business rules**.

ANOTHER EXAMPLE: Perhaps the student may not have a PHONE when they first register, so that may be an optional attribute. The organization wants to record information about phone numbers, but when the information about the student is recorded or stored, a phone number may not exist.

EXAMPLE: A mandatory field would be name. No student can register without a name.

EXAMPLE: Students, when they first register may not be assigned a course and most certainly will not have a grade yet. Those fields would be optional with respect to a student entity.

FACULTY ???? We can assume all FACULTY data is required.

The faculty may be hired, and the faculty number and name stored but no office and phone are yet available or assigned to them. Again, the organization determines the rules and in this case the organization rules say that it requires that an office and phone be assigned at the time of storing the faculty record.

**ENTITY LISTING -- after optional**

An entity list for the STUDENT TRANSCRIPT SYSTEM with the optional attributes identified would look like this:

STUDENT [STUDNO, NAME, ADDRESS, phone, crscode, crsdesc, grade]

FACULTY [FACNO, FNAME, FOFFICE, FPHONE]

**NOTATION**

**DETERMINE REPEATING ATTRIBUTES**

The next step in refining the attribute list is to determine which attributes can possibly occur more than once for an occurrence of the entity. This process is part of the **normalization** process that will be discussed later. It is introduced here so that the notation can be shown and to practice looking for such occurrences.

It is reasonable to assume that a student could have more than one phone number where they may be contacted. For this system, the organization wants to be able to store more than one phone. In that case, then our attribute list should in some way note this. We can show repeating attributes with round brackets ( ). These brackets are placed around the attribute (Phone).

The fact that the attribute is optional or mandatory has no bearing on whether it can repeat.

# REPEATING GROUPS

If two or more attributes will repeat together, then the entire group of attributes is surrounded with round brackets ( ).

EXAMPLE:

This is the case with (crscode, crsdesc, grade).

Every time there is an occurrence of course data for a student, then there will be a crscode, crsdesc and eventually a grade.

You will never have a situation where there would be 9 crscode occurrences but only 6 crsdesc occurrences.

### ENTITY LIST – now showing repeating data and repeating groups

STUDENT [STUDNO, NAME, ADDRESS, (phone), (crscode, crsdesc, grade)]

FACULTY [FACNO, FNAME, OFFICE, FPHONE]

The above is a partial or preliminary Entity Attribute list. It is NOT complete

**APPLY SAMPLE OR REAL DATA**

**-- TO THE RESULT**

Using the two sample transcripts lets see what the data would look like if we were to store it in the fashion described above in the entity list. This is often a good tool to use to see if there is anything unusual about the data

STUDENT

STUDNO NAME STREET CITY PROV POSTAL PHONE CODE DESCRIPTION GRADE

1234567 Mary Kim 123 This Street No City ON A9A9A9 8887654 PAS145 PASCAL 2.0

1234567 Mary Kim 123 This Street No City ON A9A9A9 8887654 COM223 COMMUNICATIONS 3.6

1234567 Mary Kim 123 This Street No City ON A9A9A9 8887654 MAT111 MATHEMATICS 2.4

7654321 John Doe 345 That Street My City ON X1X1X1 8950001 PAS145 PASCAL 2.0

7654321 John Doe 345 That Street My City ON X1X1X1 8950024 COM223 COMMUNICATIONS 2.7

7654321 John Doe 345 That Street My City ON X1X1X1 8950024 MAT111 MATHEMATICS 3.0

FACULTY

FACNO FNAME OFFICE FPHONE

CS245 TOM JONES 135 7229

AP142 SALLY ALI 139 7201

CS153 KRIS GOLLY 130 7255

**Can you see any problems with this data?**

**How many times have we stored the name Mary Kim?**

**Why?**

-- Each of the courses taken are different,

-- But the row of data includes name and address etc.

How many times have we stored the course description COMMUNICATIONS? Why?

-- Same type of problem -- we have many students taking the same course

**Is there any way of knowing, which courses TOM JONES, teaches?**

**Why not?**

- There is no connection between the two entities

**The problems with this design become very apparent when we load actual data into the design**.

#### NEXT MODULES DICUSS

The following modules show you how to solve these problems and others, through normalization and other steps, so that you can create database tables, and load appropriate data.

**MODULE 02 WORKSHOP**

1. For each of the first 3 cases develop an entity list using the suggested notation in this module.

Be sure to identify the optional attributes and the repeating attributes.

Be sure to write down any assumptions that you have made concerning the applications.

Suggestions:

MAYBE ALLOW TIME IN CLASS ON VET case

BREAK INTO GROUPS TO DO OTHER 2 cases