Day 2: Planning Your Database

Understanding the Relational Model, Organizing Data for It

Database Theory and Design
Tyler Peterson

International Summer School on Language Documentation and Description
Leiden University Centre for Linguistics, Leiden

July 6, 2010

Goals for Today:

A Review

- The 'Flat' database, and the DBMS
- The Database Management System (DBMS)

The Relational Database Model

- Understanding the Relational Database
- A flat table converted into realtions
- In class exercise: A mini-relational database for Gitksan

The Next Step: Designing a Relational Database

- Preliminary steps, and the "Systems Development Life Cycle"
- Planning
- Analysis & Design

Exercise and Follow-up

- In class exercise: An Mini-relational database
- The Next Step: Implementing a Relational Database
- Take-home Assignment

A common starting point: the 'Flat' database

Word	Gloss	Gram.	Morph.
hon	'fish'	N	ROOT
smax	'bear, meat'	N	ROOT
algyax̲	'language'	N	ROOT
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	'true'	A	PREFIX
siipxw	'sick, ill'	A	ROOT
wii-n̓akw	'tall'	A	STEM
wii-	'long'	A	PREFIX
n̓akw	DISTAL		ROOT
n̓akw	EVIDENTIAL		ROOT
x̲-	'consume'	V	PREFIX
iixwt	'fish'	V	ROOT
witxw	'arrive'	V	ROOT
bakw	'arrive'	V	ROOT
litsx̲xw	'read'	V	ROOT
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A 'Flat' Database of a Gitksan (Tsimshianic) word list

The Solution:

- Separate the flat database into two interacting systems:

The Solution:

- Separate the flat database into two interacting systems:
  I. Database Management System (DBMS)
  II. An application to interact with the DBMS.

The Solution:

- Separate the flat database into two interacting systems:
  I. Database Management System (DBMS)
  II. An application to interact with the DBMS.
- Applications such as MS Access and OO Base contain both of these components.

The Solution:

- Separate the flat database into two interacting systems:
  I. Database Management System (DBMS)
  II. An application to interact with the DBMS.
- Applications such as MS Access and OO Base contain both of these components.
- A Relational Database: Consists of several tables called *relations*, that are linked to each other.

The Solution:

- Separate the flat database into two interacting systems:
  I. Database Management System (DBMS)
  II. An application to interact with the DBMS.
- Applications such as MS Access and OO Base contain both of these components.
- A Relational Database: Consists of several tables called *relations*, that are linked to each other.

★: **Before we can start working on our database we need to understand how a relational database is modeled.**

The Characteristics:

- We have to rethink how we see and how we approach organizing our data, using four conceptual terms as our guide:

The Characteristics:

- We have to rethink how we see and how we approach organizing our data, using four conceptual terms as our guide:

  1. **Relations**
  2. **Entities**
  3. **Attributes**
  4. **Records**

The Characteristics:

- The **Relation**: a table with named columns.

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.
  - For the linguist: words, sentences, nasal vowels, verb roots etc.

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.
  - For the linguist: words, sentences, nasal vowels, verb roots etc.
- Each entity has a set of **Attributes**:

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.
  - For the linguist: words, sentences, nasal vowels, verb roots etc.
- Each entity has a set of **Attributes**:
  - The set of characteristics associated with the entity.

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.
  - For the linguist: words, sentences, nasal vowels, verb roots etc.
- Each entity has a set of **Attributes**:
  - The set of characteristics associated with the entity.
  - Words: gloss, grammatical and morphological categories.

The Characteristics:

- The **Relation**: a table with named columns.
- The **Entity**:
  - Defines the type of data we intend to collect.
  - For the linguist: words, sentences, nasal vowels, verb roots etc.
- Each entity has a set of **Attributes**:
  - The set of characteristics associated with the entity.
  - Words: gloss, grammatical and morphological categories.
- A **Record**: Each (unique) row in a table is a single record within the entity set (simply listing of all the related entities in a set).

A simple example

Word	Gloss	Gram.	Morph.
hon	'fish'	N	ROOT
smax	'bear, meat'	N	ROOT
algyax̱	'language'	N	ROOT
sm̓-algyax̱	Gitksan	N	STEM
sm̓-	'true'	A	PREFIX
siipxw	'sick, ill'	A	ROOT
wii-n̓akw	'tall'	A	STEM
wii-	'long'	A	PREFIX
n̓akw	DISTAL		ROOT
n̓akw	EVIDENTIAL		ROOT
x̱-	'consume'	V	PREFIX
iixwt	'fish'	V	ROOT
witxw	'arrive'	V	ROOT
bakw	'arrive'	V	ROOT
litsx̱xw	'read'	V	ROOT
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: The Entity set: Words

- Consider the familiar, standard word list. What is it in relational terms?

A simple example

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	ROOT
smax	‘bear, meat’	N	ROOT
algyax̱	‘language’	N	ROOT
sm̓-algyax̱	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	ROOT
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		ROOT
n̓akw	EVIDENTIAL		ROOT
x̱-	‘consume’	V	PREFIX
iixwt	‘fish’	V	ROOT
witxw	‘arrive’	V	ROOT
bakw	‘arrive’	V	ROOT
litsx̱xw	‘read’	V	ROOT
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: The Entity set: Words

- Consider the familiar, standard word list. What is it in relational terms?
- **An entity set of Gitksan words.**

A simple example

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	ROOT
smax	‘bear, meat’	N	ROOT
algyax̲	‘language’	N	ROOT
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	ROOT
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		ROOT
n̓akw	EVIDENTIAL		ROOT
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	ROOT
witxw	‘arrive’	V	ROOT
bakw	‘arrive’	V	ROOT
litsx̲xw	‘read’	V	ROOT
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: The Entity set: Words

- Consider the familiar, standard word list. What is it in relational terms?
- **An entity set of Gitksan words.**
  - Each Gitksan word is an entity: it occupies a unique row.

A simple example

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	ROOT
smax	‘bear, meat’	N	ROOT
algyax̲	‘language’	N	ROOT
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	ROOT
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		ROOT
n̓akw	EVIDENTIAL		ROOT
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	ROOT
witxw	‘arrive’	V	ROOT
bakw	‘arrive’	V	ROOT
litsx̲xw	‘read’	V	ROOT
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: The Entity set: Words

- Consider the familiar, standard word list. What is it in relational terms?
- **An entity set of Gitksan words.**
  - Each Gitksan word is an entity: it occupies a unique row.
  - Each row is associated with a set of attributes: **Gloss, Gram.** and **Morph.**

A simple example cont.

Gloss	**Word**	**Gram.**	**Morph.**
‘fish’	*hon*	N	**ROOT**
‘bear, meat’	*smax*	N	**ROOT**
‘language’	*algyax̱*	N	**ROOT**
Gitksan	*sm̓-algyax̱*	N	STEM
‘true’	*sm̓-*	A	PREFIX
‘sick, ill’	*siipxw*	A	**ROOT**
‘tall’	*wii-n̓akw*	A	STEM
‘long’	*wii-*	A	PREFIX
DISTAL	*n̓akw*		**ROOT**
EVIDENTIAL	*n̓akw*		**ROOT**
‘consume’	*x̱-*	V	PREFIX
‘fish’	*iixwt*	V	**ROOT**
‘arrive’	*witxw*	V	**ROOT**
‘arrive’	*bakw*	V	**ROOT**
‘read’	*litsx̱xw*	V	**ROOT**
common noun	*=hl*	Det.	ENCLITIC
proper noun	*=t*	Det.	ENCLITIC
plural noun	*=tip*	Det.	ENCLITIC
1sg	*-y̓*	Agr.	SUFFIX
2sg	*-n*	Agr.	SUFFIX
3	*-t*	Agr.	SUFFIX

Table: The Entity set: Gloss

- However, what if we’re making a dictionary?

A simple example cont.

Gloss	Word	Gram.	Morph.
‘fish’	*hon*	N	**ROOT**
‘bear, meat’	*smax*	N	**ROOT**
‘language’	*algyax̲*	N	**ROOT**
Gitksan	*sm̓-algyax̲*	N	STEM
‘true’	*sm̓-*	A	PREFIX
‘sick, ill’	*siipxw*	A	**ROOT**
‘tall’	*wii-n̓akw*	A	STEM
‘long’	*wii-*	A	PREFIX
DISTAL	*n̓akw*		**ROOT**
EVIDENTIAL	*n̓akw*		**ROOT**
‘consume’	*x̲-*	V	PREFIX
‘fish’	*iixwt*	V	**ROOT**
‘arrive’	*witxw*	V	**ROOT**
‘arrive’	*bakw*	V	**ROOT**
‘read’	*litsx̲xw*	V	**ROOT**
common noun	*=hl*	Det.	ENCLITIC
proper noun	*=t*	Det.	ENCLITIC
plural noun	*=tip*	Det.	ENCLITIC
1sg	*-y̓*	Agr.	SUFFIX
2sg	*-n*	Agr.	SUFFIX
3	*-t*	Agr.	SUFFIX

Table: The Entity set: Gloss

- However, what if we’re making a dictionary?
- We would want to also have an entity set based on the Gloss.

A simple example cont.

Gloss	Word	Gram.	Morph.
‘fish’	*hon*	N	**ROOT**
‘bear, meat’	*smax*	N	**ROOT**
‘language’	*algyax̱*	N	**ROOT**
Gitksan	*sm̓-algyax̱*	N	STEM
‘true’	*sm̓-*	A	PREFIX
‘sick, ill’	*siipxw*	A	**ROOT**
‘tall’	*wii-n̓akw*	A	STEM
‘long’	*wii-*	A	PREFIX
DISTAL	*n̓akw*		**ROOT**
EVIDENTIAL	*n̓akw*		**ROOT**
‘consume’	*x̱-*	V	PREFIX
‘fish’	*iixwt*	V	**ROOT**
‘arrive’	*witxw*	V	**ROOT**
‘arrive’	*bakw*	V	**ROOT**
‘read’	*litsx̱xw*	V	**ROOT**
common noun	*=hl*	Det.	ENCLITIC
proper noun	*=t*	Det.	ENCLITIC
plural noun	*=tip*	Det.	ENCLITIC
1sg	*-y̓*	Agr.	SUFFIX
2sg	*-n*	Agr.	SUFFIX
3	*-t*	Agr.	SUFFIX

Table: The Entity set: Gloss

- However, what if we’re making a dictionary?
- We would want to also have an entity set based on the Gloss.
- So far, we been using the attributes (the columns in our flat database) to define new relations (entity sets).

A simple example cont.

Gloss	**Word**	**Gram.**	**Morph.**
‘fish’	*hon*	N	**ROOT**
‘bear, meat’	*smax*	N	**ROOT**
‘language’	*algyax̱*	N	**ROOT**
Gitksan	*sm̓-algyax̱*	N	STEM
‘true’	*sm̓-*	A	PREFIX
‘sick, ill’	*siipxw*	A	**ROOT**
‘tall’	*wii-n̓akw*	A	STEM
‘long’	*wii-*	A	PREFIX
DISTAL	*n̓akw*		**ROOT**
EVIDENTIAL	*n̓akw*		**ROOT**
‘consume’	*x̱-*	V	PREFIX
‘fish’	*iixwt*	V	**ROOT**
‘arrive’	*witxw*	V	**ROOT**
‘arrive’	*bakw*	V	**ROOT**
‘read’	*litsx̱xw*	V	**ROOT**
common noun	*=hl*	Det.	ENCLITIC
proper noun	*=t*	Det.	ENCLITIC
plural noun	*=tip*	Det.	ENCLITIC
1sg	*-y̓*	Agr.	SUFFIX
2sg	*-n*	Agr.	SUFFIX
3	*-t*	Agr.	SUFFIX

Table: The Entity set: Gloss

- However, what if we’re making a dictionary?
- We would want to also have an entity set based on the Gloss.
- So far, we been using the attributes (the columns in our flat database) to define new relations (entity sets).
- However, it is the values of these attributes that defines entity sets as a natural class of objects.

Identifying the Relations

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	**ROOT**
smax	‘bear, meat’	N	**ROOT**
algyax̲	‘language’	N	**ROOT**
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	**ROOT**
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		**ROOT**
n̓akw	EVIDENTIAL		**ROOT**
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	**ROOT**
witxw	‘arrive’	V	**ROOT**
bakw	‘arrive’	V	**ROOT**
litsx̲xw	‘read’	V	**ROOT**
=*hl*	common noun	Det.	ENCLITIC
=*t*	proper noun	Det.	ENCLITIC
=*tip*	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A ‘Flat’ Database

- This is useful for languages data:

Identifying the Relations

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	**ROOT**
smax	‘bear, meat’	N	**ROOT**
algyax̱	‘language’	N	**ROOT**
sm̓-algyax̱	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	**ROOT**
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		**ROOT**
n̓akw	EVIDENTIAL		**ROOT**
x̱-	‘consume’	V	PREFIX
iixwt	‘fish’	V	**ROOT**
witxw	‘arrive’	V	**ROOT**
bakw	‘arrive’	V	**ROOT**
litsx̱xw	‘read’	V	**ROOT**
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A ‘Flat’ Database

- This is useful for languages data:
- What if we want to investigate different morphological root shapes? Or a list of nouns? or a list of verb roots?

Identifying the Relations

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	**ROOT**
smax	‘bear, meat’	N	**ROOT**
algyax̲	‘language’	N	**ROOT**
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	**ROOT**
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		**ROOT**
n̓akw	EVIDENTIAL		**ROOT**
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	**ROOT**
witxw	‘arrive’	V	**ROOT**
bakw	‘arrive’	V	**ROOT**
litsx̲xw	‘read’	V	**ROOT**
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A ‘Flat’ Database

- This is useful for languages data:
- What if we want to investigate different morphological root shapes? Or a list of nouns? or a list of verb roots?
- We create entity sets for these.

Identifying the Relations

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	**ROOT**
smax	‘bear, meat’	N	**ROOT**
algyax̲	‘language’	N	**ROOT**
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	**ROOT**
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		**ROOT**
n̓akw	EVIDENTIAL		**ROOT**
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	**ROOT**
witxw	‘arrive’	V	**ROOT**
bakw	‘arrive’	V	**ROOT**
litsx̲xw	‘read’	V	**ROOT**
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A ‘Flat’ Database

- This is useful for languages data:
- What if we want to investigate different morphological root shapes? Or a list of nouns? or a list of verb roots?
- We create entity sets for these.
- For example, the morphological category ‘ROOTS’ become the entity set containing everything that is ROOT, while excluding everything that isn’t.

Identifying the Relations

Word	Gloss	Gram.	Morph.
hon	‘fish’	N	**ROOT**
smax	‘bear, meat’	N	**ROOT**
algyax̲	‘language’	N	**ROOT**
sm̓-algyax̲	Gitksan	N	STEM
sm̓-	‘true’	A	PREFIX
siipxw	‘sick, ill’	A	**ROOT**
wii-n̓akw	‘tall’	A	STEM
wii-	‘long’	A	PREFIX
n̓akw	DISTAL		**ROOT**
n̓akw	EVIDENTIAL		**ROOT**
x̲-	‘consume’	V	PREFIX
iixwt	‘fish’	V	**ROOT**
witxw	‘arrive’	V	**ROOT**
bakw	‘arrive’	V	**ROOT**
litsx̲xw	‘read’	V	**ROOT**
=hl	common noun	Det.	ENCLITIC
=t	proper noun	Det.	ENCLITIC
=tip	plural noun	Det.	ENCLITIC
-y̓	1sg	Agr.	SUFFIX
-n	2sg	Agr.	SUFFIX
-t	3	Agr.	SUFFIX

Table: A ‘Flat’ Database

Word	Gloss	Gram.
hon	‘fish’	N
smax	‘bear, meat’	N
algyax̲	‘language’	N
siipxw	‘sick, ill’	A
n̓akw	DISTAL	
n̓akw	EVIDENTIAL	
iixwt	‘fish’	V
witxw	‘arrive’	V
bakw	‘arrive’	V
litsx̲xw	‘read’	V

Table: The Entity set: ROOTS

A mini-relational database for Gitksan

- Form groups of 3 and work out how many relations (tables) there are in the Gitksan dataset on the handout.

A mini-relational database for Gitksan

- Form groups of 3 and work out how many relations (tables) there are in the Gitksan dataset on the handout.
- (Hint: Be as exhaustive as possible, including trying out ALL the attributes and features...)

Discussion

- In addition to Word, Gloss, and Roots, the Gitksan flat DB can be organized into at least X more unique entity sets:

Word	Gloss	Morph.
hon	‘fish’	ROOT
smax	‘bear, meat’	ROOT
algyax̲	‘language’	ROOT
sm̓-algyax̲	Gitksan	STEM

Table: The Entity set: NOUNS

Word	Gloss	Morph.
x̲-	‘consume’	PREFIX
iixwt	‘fish’	ROOT
witxw	‘arrive’	ROOT
bakw	‘arrive’	ROOT
litsx̲xw	‘read’	ROOT

Table: The Entity set: VERBS

Word	Gloss	Morph.
sm̓-	‘true’	PREFIX
siipxw	‘sick, ill’	ROOT
wii-n̓akw	‘tall’	STEM
wii-	‘long’	PREFIX

Table: The Entity set: ADJ.

Word	Gloss	Morph.
=hl	common noun	ENCLITIC
=t	proper noun	ENCLITIC
=tip	plural noun	ENCLITIC

Table: The Entity set: DET.

Discussion cont.

Word	Gloss	Morph.
-y̓	1sg	SUFFIX
-n	2sg	SUFFIX
-t	3	SUFFIX

Table: The Entity set: AGR

Word	Gloss	Gram.
sm̓-algyax̲	Gitksan	N
wii-n̓akw	‘tall’	A

Table: The Entity set: STEM

Word	Gloss	Gram.
sm̓-	‘true’	A
wii-	‘long’	A
x̲-	‘consume’	V

Table: The Entity set: PREFIX

Word	Gloss	Gram.
=hl	common noun	Det.
=t	proper noun	Det.
=tip	plural noun	Det.

Table: The Entity set: ENCLITIC

Discussion cont.

Word	Gloss	Gram.
-y̓	1sg	Agr.
-n	2sg	Agr.
-t	3	Agr.

Table: The Entity set: SUFFIX

Word	Gloss
iixwt	'fish'
witxw	'arrive'
bakw	'arrive'
litsx̱xw	'read'

Table: The Entity set: V ROOTS

Word	Gloss
hon	'fish'
smax	'bear, meat'
algyax̱	'language'

Table: The Entity set: N ROOTS

Word	Gloss
siipxw	'sick, ill'

Table: The Entity set: A ROOTS

- And more...

Designing a Relational Database

- In the next section we will take the first steps on designing a relational database.

Designing a Relational Database

- In the next section we will take the first steps on designing a relational database.
  - Examining your data and determining your goals: turning a flat database into a relational one.

Designing a Relational Database

- In the next section we will take the first steps on designing a relational database.
  - Examining your data and determining your goals: turning a flat database into a relational one.
  - The "Systems Development Life Cycle": the planning, design and implementation of your database in FileMaker PRO and MS Access.

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).
- Thus, relational databases consist of one or more tables which are related to each other based on sharing common entities.

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).
- Thus, relational databases consist of one or more tables which are related to each other based on sharing common entities.
- These tables can be ‘joined’ by the database software when making **queries**.

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).
- Thus, relational databases consist of one or more tables which are related to each other based on sharing common entities.
- These tables can be ‘joined’ by the database software when making **queries**.
- Each table

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).
- Thus, relational databases consist of one or more tables which are related to each other based on sharing common entities.
- These tables can be ‘joined’ by the database software when making **queries**.
- Each table
  - is actually an **entity set**, or one ‘subject’ of the database, i.e. Gitksan words, ROOTS, or Grammatical Categories etc.

Pulling the Conceptual Pieces together

- A flat database can be re-organized into a collection of unique tables (also called **relations**).
- Thus, relational databases consist of one or more tables which are related to each other based on sharing common entities.
- These tables can be 'joined' by the database software when making **queries**.
- Each table
  - is actually an **entity set**, or one 'subject' of the database, i.e. Gitksan words, ROOTS, or Grammatical Categories etc.
  - has a set of attributes: these are typically your column headings which encode that attributes of the individual entities within the set.

Some Entity sets

Word	Gloss	Morph.
hon	‘fish’	**ROOT**
smax	‘bear, meat’	**ROOT**
algyax̱	‘language’	**ROOT**
sm̓-algyax̱	Gitksan	STEM

Table: The Entity set: Nouns

Word	Gloss	Morph.
x̱-	‘consume’	PREFIX
iixwt	‘fish’	**ROOT**
witxw	‘arrive’	**ROOT**
bakw	‘arrive’	**ROOT**
litsx̱xw	‘read’	**ROOT**

Table: The Entity set: Verbs

Word	Gloss	Gram.
hon	‘fish’	N
smax	‘bear, meat’	N
algyax̱	‘language’	N
siipxw	‘sick, ill’	A
n̓akw	DISTAL	
n̓akw	EVIDENTIAL	
iixwt	‘fish’	V
witxw	‘arrive’	V
bakw	‘arrive’	V
litsx̱xw	‘read’	V

Table: The Entity set: ROOTS

The “Systems Development Life Cycle”

- A concept from database design theory: the *Systems Development Life Cycle* (SDLS).
- The SDLS is a useful set of (re)iterative conceptual phases:

	Phase	**Action**
1.	*Planning*	What is the the purpose of the database?
2.	*Analysis*	Assessing and organizing the data.
3.	*Design*	Logical design of the database.
4.	*Implementation*	MS Access, FMP, OO Base etc.
5.	*Maintenance*	Evaluation, maintenance, further development

Preliminary steps

- First step: Start with a pencil and paper.

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It’s overall scope and objectives.
  - To generate word lists (i.e. lexicographic)

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.
  - To generate word lists (i.e. lexicographic)
  - An organization meta-data linked to language data

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.
  - To generate word lists (i.e. lexicographic)
  - An organization meta-data linked to language data
  - Archiving

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.
    - To generate word lists (i.e. lexicographic)
    - An organization meta-data linked to language data
    - Archiving
    - Analysis

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.
  - To generate word lists (i.e. lexicographic)
  - An organization meta-data linked to language data
  - Archiving
  - Analysis
  - Etc.

Preliminary steps

- First step: Start with a pencil and paper.
- What is the purpose of your database? It's overall scope and objectives.
    - To generate word lists (i.e. lexicographic)
    - An organization meta-data linked to language data
    - Archiving
    - Analysis
    - Etc.
- **There is no single, out-of-the-box solution:** think of all the things you could possible document about a language, and all the different ways to look at it and organize it – that's how many potentially different kinds of databases you could make.

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:
  - Look at what the potential entities in the database are about rather than the *conclusions* that you want to find.

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:
  - Look at what the potential entities in the database are about rather than the *conclusions* that you want to find.
  - Think about the entities (and thus the data) separately from practical considerations, such as who is going to enter the data, forms, encoding etc.

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:
  - Look at what the potential entities in the database are about rather than the *conclusions* that you want to find.
  - Think about the entities (and thus the data) separately from practical considerations, such as who is going to enter the data, forms, encoding etc.
  - Think about the subject independently of any particular database software, or even of computing at all.

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:
  - Look at what the potential entities in the database are about rather than the *conclusions* that you want to find.
  - Think about the entities (and thus the data) separately from practical considerations, such as who is going to enter the data, forms, encoding etc.
  - Think about the subject independently of any particular database software, or even of computing at all.
  - Avoid getting confused between the overview of the data and details of implementation.

Some Heuristic Steps

- In designing a database it is useful to start at a very high level of generalization:
    - Look at what the potential entities in the database are about rather than the *conclusions* that you want to find.
    - Think about the entities (and thus the data) separately from practical considerations, such as who is going to enter the data, forms, encoding etc.
    - Think about the subject independently of any particular database software, or even of computing at all.
    - Avoid getting confused between the overview of the data and details of implementation.
- Work through the construction of an appropriate diagram for your database: This diagram is the basis for the tables and fields.

Dos and Don’ts

The "Entity-Attribute-Relationship Diagram"

- A simple, common and useful model/technique for mapping out the logical design of a relational database: the *Entity-(Attribute-)Relationship model*.

The "Entity-Attribute-Relationship Diagram"

- A simple, common and useful model/technique for mapping out the logical design of a relational database: the *Entity-(Attribute-)Relationship model*.
- A relatively simple graphic representation of complex, real-world data structures.

The "Entity-Attribute-Relationship Diagram"

- A simple, common and useful model/technique for mapping out the logical design of a relational database: the *Entity-(Attribute-)Relationship model*.
- A relatively simple graphic representation of complex, real-world data structures.
- A theoretical modeling tool which can be implemented in any relational database application (MS Access, OO Base etc.)

The Entities

- **Entities** are the things that hold particular interest for you in your database – you can think of them as the ‘subjects’ to be covered.

The Entities

- **Entities** are the things that hold particular interest for you in your database – you can think of them as the 'subjects' to be covered.
- They are a classification of things and should have a precise definition.

The Entities

- **Entities** are the things that hold particular interest for you in your database – you can think of them as the 'subjects' to be covered.
- They are a classification of things and should have a precise definition.
- It is important to identify your entities because they usually end up being the tables in the database.

The Entities

- **Entities** are the things that hold particular interest for you in your database – you can think of them as the 'subjects' to be covered.
- They are a classification of things and should have a precise definition.
- It is important to identify your entities because they usually end up being the tables in the database.
- It may not be immediately obvious what the entities might be in your dataset.

Decide on what entities are included the database

- The best approach: take a sheet of paper and sketch each out potential entities, putting a name for each one in the box.

Decide on what entities are included the database

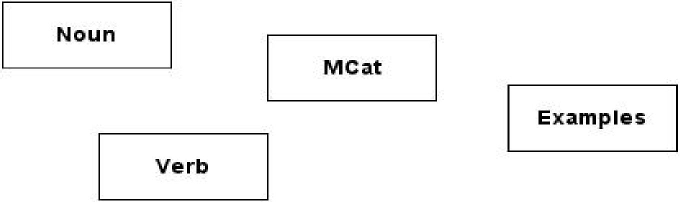
- The best approach: take a sheet of paper and sketch each out potential entities, putting a name for each one in the box.
- If you have a complex set of data, choose the most important entities to begin with or you will be overwhelmed.

Decide on what entities are included the database

- The best approach: take a sheet of paper and sketch each out potential entities, putting a name for each one in the box.
- If you have a complex set of data, choose the most important entities to begin with or you will be overwhelmed.
- An entity is represented diagrammatically by a box with a name written in the singular.

Decide on what entities are included the database

- The best approach: take a sheet of paper and sketch each out potential entities, putting a name for each one in the box.
- If you have a complex set of data, choose the most important entities to begin with or you will be overwhelmed.
- An entity is represented diagrammatically by a box with a name written in the singular.



Noun

MCat

Examples

Verb

The Attributes

- **Attributes** are the details about an entity: they are the individual relevant properties of things we want to know about.

Word	Gloss	MCat.
hon	'fish'	**ROOT**
smax	'bear, meat'	**ROOT**
algyax̲	'language'	**ROOT**
sm̓-algyax̲	Gitksan	STEM

Table: The Entity set: Nouns

Word	Gloss	MCat.
x̲-	'consume'	PREFIX
iixwt	'fish'	**ROOT**
witxw	'arrive'	**ROOT**
bakw	'arrive'	**ROOT**
litsx̲xw	'read'	**ROOT**

Table: The Entity set: Verbs

The Attributes

- **Attributes** are the details about an entity: they are the individual relevant properties of things we want to know about.
- The entity of Gitksan Nouns or Verbs has the attributes: the Gitksan word, its gloss, and its morphological category.

Word	Gloss	MCat.
hon	'fish'	**ROOT**
smax	'bear, meat'	**ROOT**
algyax̱	'language'	**ROOT**
sm̓-algyax̱	Gitksan	STEM

Table: The Entity set: Nouns

Word	Gloss	MCat.
x̱-	'consume'	PREFIX
iixwt	'fish'	**ROOT**
witxw	'arrive'	**ROOT**
bakw	'arrive'	**ROOT**
litsx̱xw	'read'	**ROOT**

Table: The Entity set: Verbs

The Attributes

- **Attributes** are the details about an entity: they are the individual relevant properties of things we want to know about.
- The entity of Gitksan Nouns or Verbs has the attributes: the Gitksan word, its gloss, and its morphological category.
- Attributes represent the data that characterizes the entities – and they usually become the rows in the tables.

Word	Gloss	MCat.
hon	‘fish’	**ROOT**
smax	‘bear, meat’	**ROOT**
algyax̲	‘language’	**ROOT**
sm̓-algyax̲	Gitksan	STEM

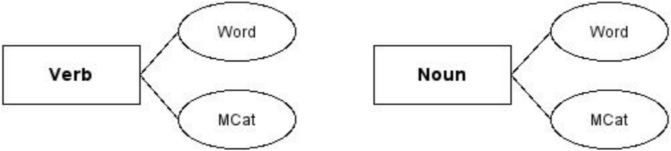
Table: The Entity set: Nouns

Word	Gloss	MCat.
x̲-	‘consume’	PREFIX
iixwt	‘fish’	**ROOT**
witxw	‘arrive’	**ROOT**
bakw	‘arrive’	**ROOT**
litsx̲xw	‘read’	**ROOT**

Table: The Entity set: Verbs

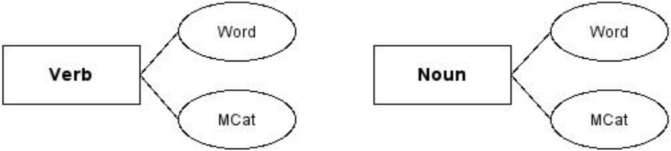
Identifying the attributes of entities

- In an ER diagram, attributes are represented by ovals that are associated with the boxed entity by drawing a line connecting them.



Identifying the attributes of entities

- In an ER diagram, attributes are represented by ovals that are associated with the boxed entity by drawing a line connecting them.



- An alternative notation: VERB(Word, MCat)

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
    - The word

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
    - The word
    - Mass/count
    - Alienability

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category
- Let's add an entity set of verbs to our database:

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
    - The word
    - Mass/count
    - Alienability
    - Morphological category
- Let's add an entity set of verbs to our database:
    - The word

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category
- Let's add an entity set of verbs to our database:
  - The word
  - Argument structure

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category
- Let's add an entity set of verbs to our database:
  - The word
  - Argument structure
  - Event structure

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category
- Let's add an entity set of verbs to our database:
  - The word
  - Argument structure
  - Event structure
  - Morphological category

Identifying the attributes of entities cont.

- Think about a simple lexicographic/field database containing the entity set of Nouns: what kinds and how many attributes would you associate to this entity set?
  - The word
  - Mass/count
  - Alienability
  - Morphological category
- Let's add an entity set of verbs to our database:
  - The word
  - Argument structure
  - Event structure
  - Morphological category
- The other grammatical categories will also have their own unique attributes (Adjective: physical, colour, etc.).

Identifying the attributes of entities cont.

- It is often easier to identify attributes than entities: these usually exists as columns in a pre-existing database.

Identifying the attributes of entities cont.

- It is often easier to identify attributes than entities: these usually exists as columns in a pre-existing database.
- However, attributes and entities can be easily confused because on different occasions the same items may be treated in different ways.

Identifying the attributes of entities cont.

- It is often easier to identify attributes than entities: these usually exists as columns in a pre-existing database.
- However, attributes and entities can be easily confused because on different occasions the same items may be treated in different ways.
- An attribute becomes an entity when it has significance in its own right, with its own relationships and attributes.

Identifying the attributes of entities cont.

- It is often easier to identify attributes than entities: these usually exists as columns in a pre-existing database.
- However, attributes and entities can be easily confused because on different occasions the same items may be treated in different ways.
- An attribute becomes an entity when it has significance in its own right, with its own relationships and attributes.

Identifying the attributes of entities cont.

- It is often easier to identify attributes than entities: these usually exists as columns in a pre-existing database.
- However, attributes and entities can be easily confused because on different occasions the same items may be treated in different ways.
- An attribute becomes an entity when it has significance in its own right, with its own relationships and attributes.
- Make sure that your definitions apply equally accurately to every possible instance – not just the normal case.

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:

Identifying the attributes of entities and the ‘primary key’

- Two important considerations when identifying attributes:
  - Try and identify how the attributes can be related to one another by a unique ‘key’ – called **the primary key**.

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:
  - Try and identify how the attributes can be related to one another by a unique 'key' – called **the primary key**.
  - Identify **Attribute Domains:**

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:
  - Try and identify how the attributes can be related to one another by a unique 'key' – called **the primary key**.
  - Identify **Attribute Domains:**
    - **Closed class:** MCat = {ROOT, STEM, AFFIX}

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:
  - Try and identify how the attributes can be related to one another by a unique 'key' – called **the primary key**.
  - Identify **Attribute Domains:**
    - **Closed class:** MCat = {ROOT, STEM, AFFIX}
    - **Open class:** Gloss = {tree, bear, taste, is tall ...}

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:
    - Try and identify how the attributes can be related to one another by a unique 'key' – called **the primary key**.
    - Identify **Attribute Domains:**
        - **Closed class:** MCat = {ROOT, STEM, AFFIX}
        - **Open class:** Gloss = {tree, bear, taste, is tall ...}
- Primary keys are usually have an Open class (and in an actual database, they must be unique).

Identifying the attributes of entities and the ‘primary key’

- Two important considerations when identifying attributes:
  - Try and identify how the attributes can be related to one another by a unique ‘key’ – called **the primary key**.
  - Identify **Attribute Domains:**
    - **Closed class:** MCat = {ROOT, STEM, AFFIX}
    - **Open class:** Gloss = {tree, bear, taste, is tall ...}
- Primary keys are usually have an Open class (and in an actual database, they must be unique).
- Non-primary keys usually (but not always) are closed class.

Identifying the attributes of entities and the 'primary key'

- Two important considerations when identifying attributes:
    - Try and identify how the attributes can be related to one another by a unique 'key' – called **the primary key**.
    - Identify **Attribute Domains:**
        - **Closed class:** MCat = {ROOT, STEM, AFFIX}
        - **Open class:** Gloss = {tree, bear, taste, is tall ...}
- Primary keys are usually have an Open class (and in an actual database, they must be unique).
- Non-primary keys usually (but not always) are closed class.
- We will see how determining attribute domains helps in determining relationships and adds power to queries.

Identifying the attributes of entities and the 'primary key'

- Entities may share several attributes, but in a language database they usually the word or gloss in common, which will be the primary key (underlined in verbatim notation):

Identifying the attributes of entities and the 'primary key'

- Entities may share several attributes, but in a language database they usually the word or gloss in common, which will be the primary key (underlined in verbatim notation):
  - NOUN(Gitksan, Mass/count, Alienability, MCat, Consultant, Example, Notes, CRef)

Identifying the attributes of entities and the 'primary key'

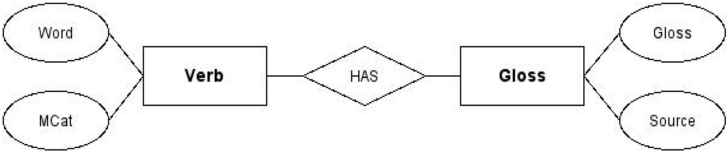
- Entities may share several attributes, but in a language database they usually the word or gloss in common, which will be the primary key (underlined in verbatim notation):
  - NOUN(Gitksan, Mass/count, Alienability, MCat, Consultant, Example, Notes, CRef)
  - VERB(Gitksan, Argument, Event, MCat, Consultant, Example, Notes, CRef)

The Relationships

- A *relationship* (not to be confused with a **relation**) is a meaningful association between two entities.

The Relationships

- A *relationship* (not to be confused with a **relation**) is a meaningful association between two entities.
- It is represented by a diamond and lines that join two entity boxes, along with a label that names the kind of relationship.



The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as ‘has’.
- For example:

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as ‘has’.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.
- But what does this relationship actually mean?

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.
- But what does this relationship actually mean?
- So far,

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.
- But what does this relationship actually mean?
- So far,
  - Every Gitksan word has a gloss,

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.
- But what does this relationship actually mean?
- So far,
  - Every Gitksan word has a gloss,
  - Every gloss has a Gitksan word, as represented by the diagram.

The Relationships cont.

- Use a simple word that encapsulates the relationship you see between the entities.
- For a language database, these are usually as simple as 'has'.
- For example:
  - **Entity 1:** GITKSAN
  - **Entity 2:** GLOSS
  - **Relationship:** between GITKSAN (words) and GLOSS.
- But what does this relationship actually mean?
- So far,
  - Every Gitksan word has a gloss,
  - Every gloss has a Gitksan word, as represented by the diagram.
- Is this always desirable?

How entities are related to one another: **Connectivity**

★ **One-to-one:** one entity can be related to only one other entity (uncommon).

How entities are related to one another: **Connectivity**

★ **One-to-one:** one entity can be related to only one other entity (uncommon).

- Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.

How entities are related to one another: **Connectivity**

- ★ **One-to-one:** one entity can be related to only one other entity (uncommon).
    - Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.
- ★ **One-to-many:** one entity can be related to several other entities.

How entities are related to one another: **Connectivity**

- ★ **One-to-one:** one entity can be related to only one other entity (uncommon).
  - Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.
- ★ **One-to-many:** one entity can be related to several other entities.
  - Every Gitksan word can have more that one gloss, or, every gloss can have more than one Gitksan word.

How entities are related to one another: **Connectivity**

- **One-to-one:** one entity can be related to only one other entity (uncommon).
  - Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.
- **One-to-many:** one entity can be related to several other entities.
  - Every Gitksan word can have more that one gloss, or, every gloss can have more than one Gitksan word.
- **Many-to-many:** many entities can be related to many other entities.

How entities are related to one another: **Connectivity**

- **One-to-one:** one entity can be related to only one other entity (uncommon).
  - Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.
- **One-to-many:** one entity can be related to several other entities.
  - Every Gitksan word can have more that one gloss, or, every gloss can have more than one Gitksan word.
- **Many-to-many:** many entities can be related to many other entities.
  - Any ideas?

How entities are related to one another: **Connectivity**

- ★ **One-to-one:** one entity can be related to only one other entity (uncommon).
  - Every Gitksan word has one meaning (gloss), and every gloss has one Gitksan word.
- ★ **One-to-many:** one entity can be related to several other entities.
  - Every Gitksan word can have more that one gloss, or, every gloss can have more than one Gitksan word.
- ★ **Many-to-many:** many entities can be related to many other entities.
  - Any ideas?
  - Several consultants can give more than one example sentence of a word.

How entities are related to one another: **Connectivity**

- **One-to-many:** one entity can be related to several other entities: A Gitksan word can have more that one gloss.

How entities are related to one another: **Connectivity**

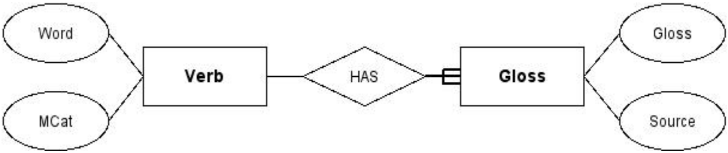
- **One-to-many:** one entity can be related to several other entities: A Gitksan word can have more that one gloss.
    - *wilp* ‘house (physical), clan’

How entities are related to one another: **Connectivity**

- **One-to-many:** one entity can be related to several other entities: A Gitksan word can have more that one gloss.
  - *wilp* ‘house (physical), clan’
  - *n̓akw* ‘evidential, spatial/temporal distal’

How entities are related to one another: **Connectivity**

- **One-to-many:** one entity can be related to several other entities: A Gitksan word can have more that one gloss.
  - *wilp* ‘house (physical), clan’
  - *n̓akw* ‘evidential, spatial/temporal distal’
- In an ER diagram, this represented by a little ‘pitchfork’ at the ‘many side’ of the relationship:



More heuristics:

- Be willing to alter the diagram and reposition the boxes and lines (i.e. use a pencil).

More heuristics:

- Be willing to alter the diagram and reposition the boxes and lines (i.e. use a pencil).
- Simplify by concentrating on the important entities – the rest can be added as your goals become clearer.

More heuristics:

- Be willing to alter the diagram and reposition the boxes and lines (i.e. use a pencil).
- Simplify by concentrating on the important entities – the rest can be added as your goals become clearer.
- There may well be more than one solution when the problem is a complex one.

More heuristics:

- Be willing to alter the diagram and reposition the boxes and lines (i.e. use a pencil).
- Simplify by concentrating on the important entities – the rest can be added as your goals become clearer.
- There may well be more than one solution when the problem is a complex one.
- Don't go crazy with relationships: they can be difficult to implement and can actually end up making your database overly-restrictive and difficult to query.

Choice 1: A mini-ER model for Gitksan

1. Form groups of 3 and work the kinds of attributes you would assign to the following entity sets:

Choice 1: A mini-ER model for Gitksan

1. Form groups of 3 and work the kinds of attributes you would assign to the following entity sets:
   - VERB
   - PRONOUN
   - DETERMINER
   - GLOSS
   - EXAMPLE

Choice 1: A mini-ER model for Gitksan

1. Form groups of 3 and work the kinds of attributes you would assign to the following entity sets:
   - VERB
   - PRONOUN
   - DETERMINER
   - GLOSS
   - EXAMPLE
2. Identity a primary key and how these entity sets might be related.

Choice 1: A mini-ER model for Gitksan

1. Form groups of 3 and work the kinds of attributes you would assign to the following entity sets:
   - VERB
   - PRONOUN
   - DETERMINER
   - GLOSS
   - EXAMPLE
2. Identity a primary key and how these entity sets might be related.

★ **Remember: there is no single, correct solution!**

Choice 2: A mini-ER model for your own language data

1. On your own (or in a group if you like) take a fragment of your own language data and work the kinds of attributes you would assign to the entity sets you've identified:

Choice 2: A mini-ER model for your own language data

1. On your own (or in a group if you like) take a fragment of your own language data and work the kinds of attributes you would assign to the entity sets you've identified:
2. Identity a primary key and how these entity sets might be related.

Choice 2: A mini-ER model for your own language data

1. On your own (or in a group if you like) take a fragment of your own language data and work the kinds of attributes you would assign to the entity sets you've identified:
2. Identity a primary key and how these entity sets might be related.

★ **Remember: there is no single, correct solution!**

Implementation in OO Base

- In the next section we will take the first steps on implementing a possible version of a relational database for our Gitksan data (and, hopefully, your own data).

Implementation in OO Base

- In the next section we will take the first steps on implementing a possible version of a relational database for our Gitksan data (and, hopefully, your own data).
  - Translating the ER diagram into Base tables

Implementation in OO Base

- In the next section we will take the first steps on implementing a possible version of a relational database for our Gitksan data (and, hopefully, your own data).
  - Translating the ER diagram into Base tables
  - Understanding the field and values associated with the attribute columns.

Implementation in OO Base

- In the next section we will take the first steps on implementing a possible version of a relational database for our Gitksan data (and, hopefully, your own data).
  - Translating the ER diagram into Base tables
  - Understanding the field and values associated with the attribute columns.
  - Implementing the relationships.

Implementation in OO Base

- In the next section we will take the first steps on implementing a possible version of a relational database for our Gitksan data (and, hopefully, your own data).
    - Translating the ER diagram into Base tables
    - Understanding the field and values associated with the attribute columns.
    - Implementing the relationships.
    - Bring the tables together: Basic querying in Base.

A relational model (ER diagram) for your own data

- Take a look at relational tables (entity sets) you made for your data from last session:

A relational model (ER diagram) for your own data

- Take a look at relational tables (entity sets) you made for your data from last session:
    - Take these entity sets and enrich them with relevant additional attributes.

A relational model (ER diagram) for your own data

- Take a look at relational tables (entity sets) you made for your data from last session:
  - Take these entity sets and enrich them with relevant additional attributes.
  - Sketch these into a simple ER diagram.

A relational model (ER diagram) for your own data

- Take a look at relational tables (entity sets) you made for your data from last session:
  - Take these entity sets and enrich them with relevant additional attributes.
  - Sketch these into a simple ER diagram.
  - Identify the relationships that connect them. (tip: don't overcomplicate these! Find an attribute they share in common.)