China Biographical Database Project (CBDB)

# T'ang Studies Society Workshop on the China Biographical Database

Harvard University
August 22-23, 2013

Sponsored by the T'ang Studies Society

China Biographical Database Project (CBDB)

Session Two:

# **The Structure and Functions of the China Biographical Database**, with Query Examples

Michael A. Fuller

# Part One: Structuring Entities in CBDB

This morning we talked about database design in general.

In this session, we are going to look at CBDB in particular.

CBDB tracks the following entities:

- People
- Places
- Offices
- Kinship
- Social Associations
- Social Distinctiveness
- Social Institutions
- Texts
- Ethnicity

I am going to go over how CBDB represents these entities as tables in some detail for two key reasons:

First, one of the central issues you will need to think about tomorrow is how to **take advantage of the entities and the table in CBDB** in your projects in your projects.

Second, the details of how we designed the tables in CBDB (especially the problems of the limits and complexity of data) may be of some help as you **think through the design of your own data**.

In order to think about **how to create and structure entities**, you will need some understanding of how CBDB has approached them.

Please open CBDB on your computer and look for the following tables:

| Entity | Table |
|---|---|
| People | BIOG_MAIN |
| Places | ADDR_CODES |
| Offices | OFFICE_CODES |
| Kinship | KINSHIP_CODES |
| Social Associations | ASSOC_CODES |
| Social Distinctiveness | STATUS_CODES |
| Social Institutions | SOCIAL INSTITUTION_CODES |
| Texts | TEXT_CODES |
| Ethnicity | ETHNICITY_TRIBE_CODES |

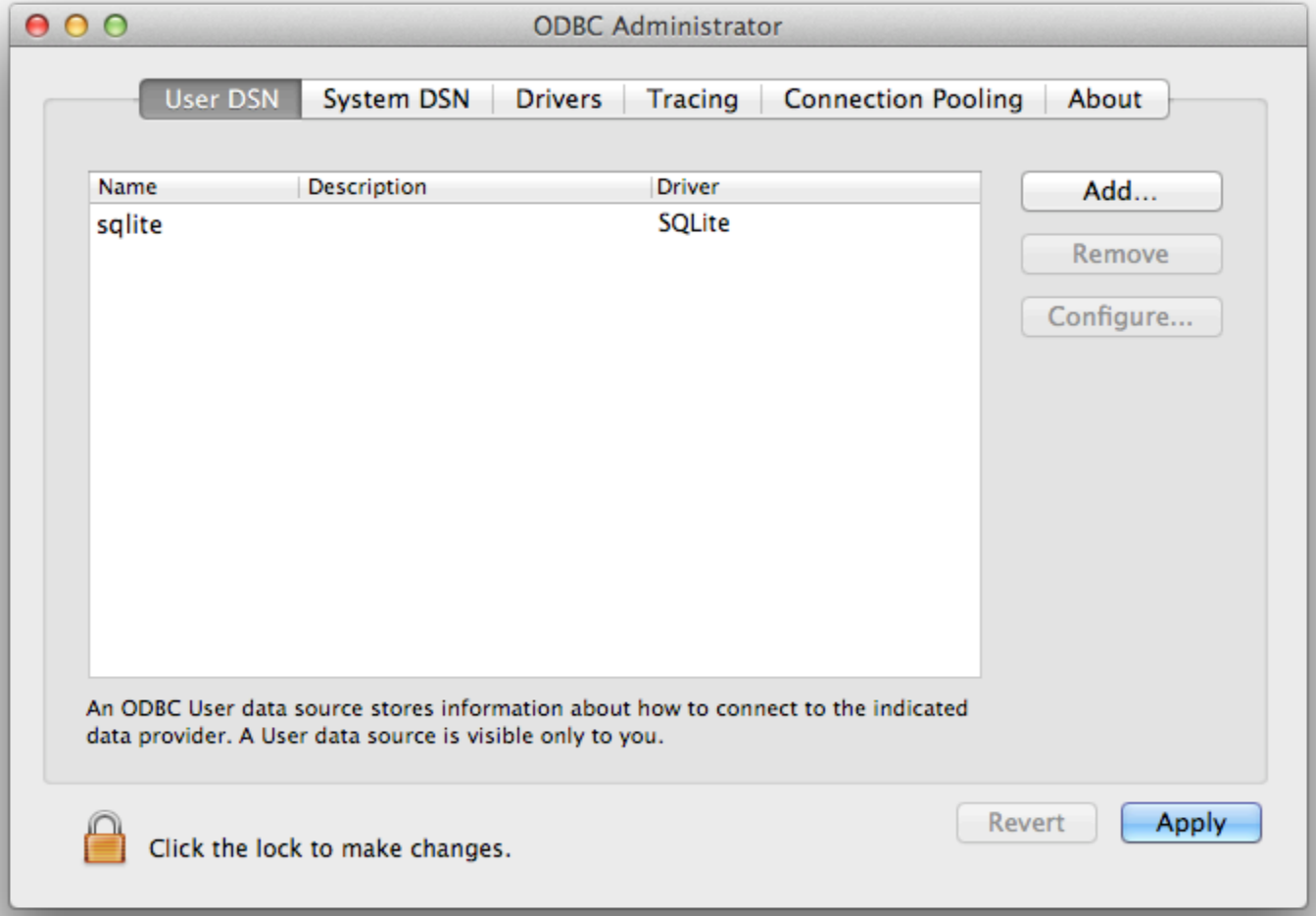
For those of you **using Macs**, we need a brief detour to set up the usage of the files.

I hope everyone has installed:

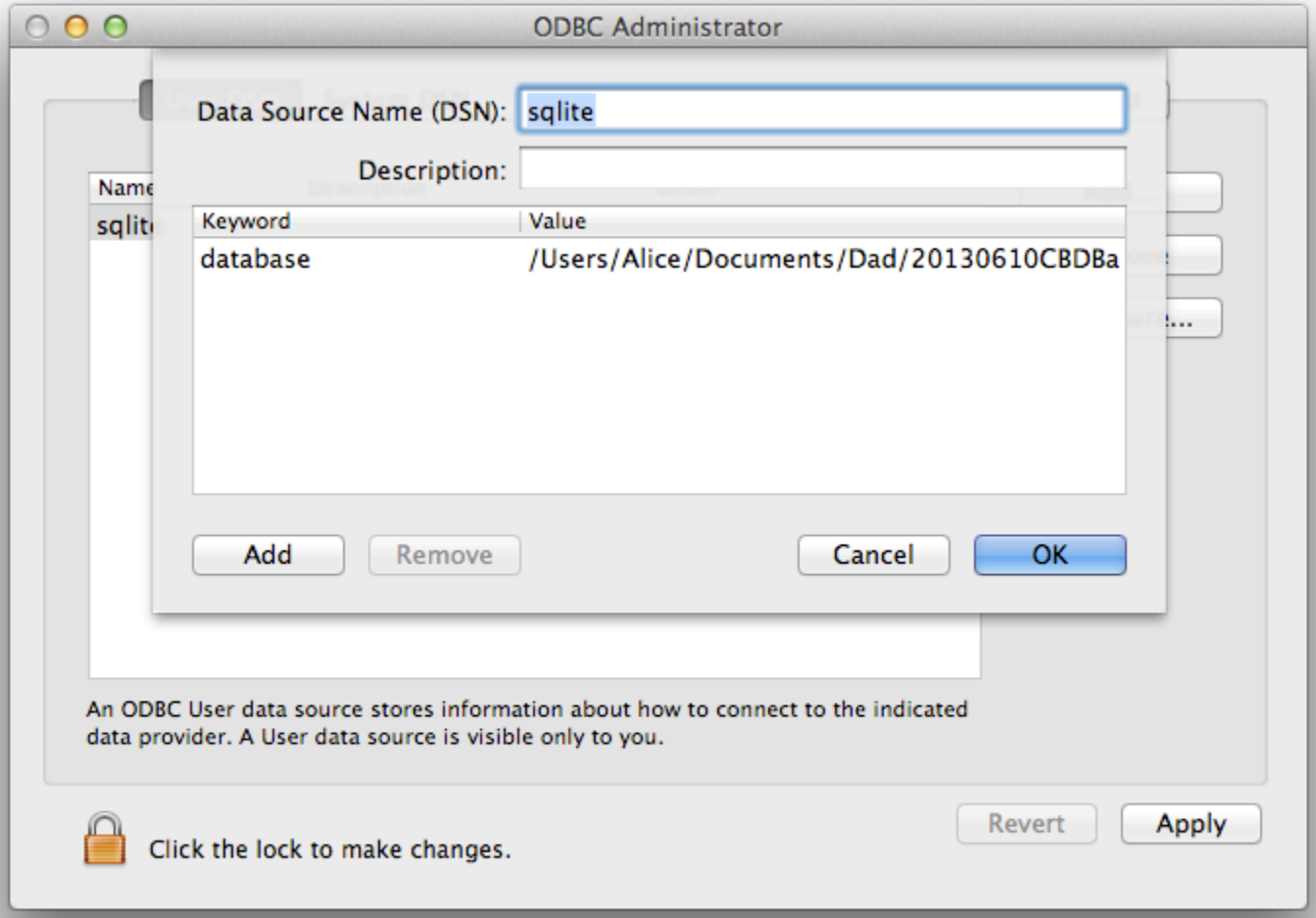
1. OpenOffice Base
2. The Mac ODBC Administrator program
3. The ODBC driver for SQLite
4. The two versions of the CBDB SQLite databases
   CBDBCore.db
   20130818CBDBao.db

Now, go into Utilities in Finder and open ODBC Administrator

# Go to User DSN and Add “CBDBFull” as an SQLite DSN



Next (and this is tedious), add the keyword “database” with the value of the full path and name of 20130818CBDBao.db



Now launch OpenOffice Base:

Choose the third option, “Connect to an existing database”

Select CBDBFull.

Save the odb file wherever you like.

Base opens, by default with the Forms.  Click on Tables.

| **Entity** | Table |
| --- | --- |
| **People** | **BIOG_MAIN** |

| **Fields** | (not all of them) |
| --- | --- |
| **c_personid** | the CBDB ID for the person |
| c_name | |
| c_name_chn | |
| c_index_year | a year used in searches |
| c_female | yes/no |
| c_ethnicity_code | |
| c_household_status_code | |
| c_birthyear | |
| c_by_nh_code | a code for the 年號 |
| c_by_nh_year | |
| c_by_range | “before/during/after” |

| Entity | Table |
| --- | --- |
| **People** | **BIOG_MAIN** |

In BIOG_MAIN some fields are codes that refer to other tables.

**ZZZ_BIOG_MAIN**, in contrast, fills in the values of these codes by using the information from the code tables. Repeating the text descriptions **violates normalization**, i.e., makes the table "**denormalized**" but easy for humans to use. For example:

| **BIOG_MAIN** | **ZZZ_BIOG_MAIN** |
| --- | --- |
| c_ethnicity_code | c_ethnicity_code |
| | c_ethnicity_chn (from ETHNICITY_TRIBES_CODES) |
| | c_ethnicity_rmn (from ETHNICITY_TRIBES_CODES) |

| Entity | Table | Denormalized Table |
|---|---|---|
| **People** | **BIOG_MAIN** | **ZZZ_BIOG_MAIN** |

BIOG_MAIN also uses a strategy to **represent uncertainty** used throughout CBDB:

- sometimes texts give us broad information about dates, like "生於開元間."
- The data is useful, if this is all we have, and we record it using a combination of 年號 and "range" information:

c_by_nh_code (375 = 開元)
c_by_nh_year (NULL)
c_by_range (0 = "during")

## Entity

**People**

In addition to the data recorded in BIOG_MAIN, CBDB also records **names**, an additional type of data concerning **People** that is not related to other entities.

Because one person may have many names, it requires a separate table:

ALTNAME_DATA (supported by the codes of ALTNAME_CODES)

And its denormalized version:

ZZZ_ALTNAME_DATA

## Entity

**People**

How **People** become eligible for **entry into Office** is not exactly an entity, but it is important. One person also may qualify for office through various distinct means (*yin* privilege, the 進士 examination followed by the 博學鴻詞, etc.)

Thus CBDB has a pair of tables to track entry into government:

ENTRY_CODES
ENTRY_DATA

## Entity

**People**

First, ENTRY_CODES (please open)

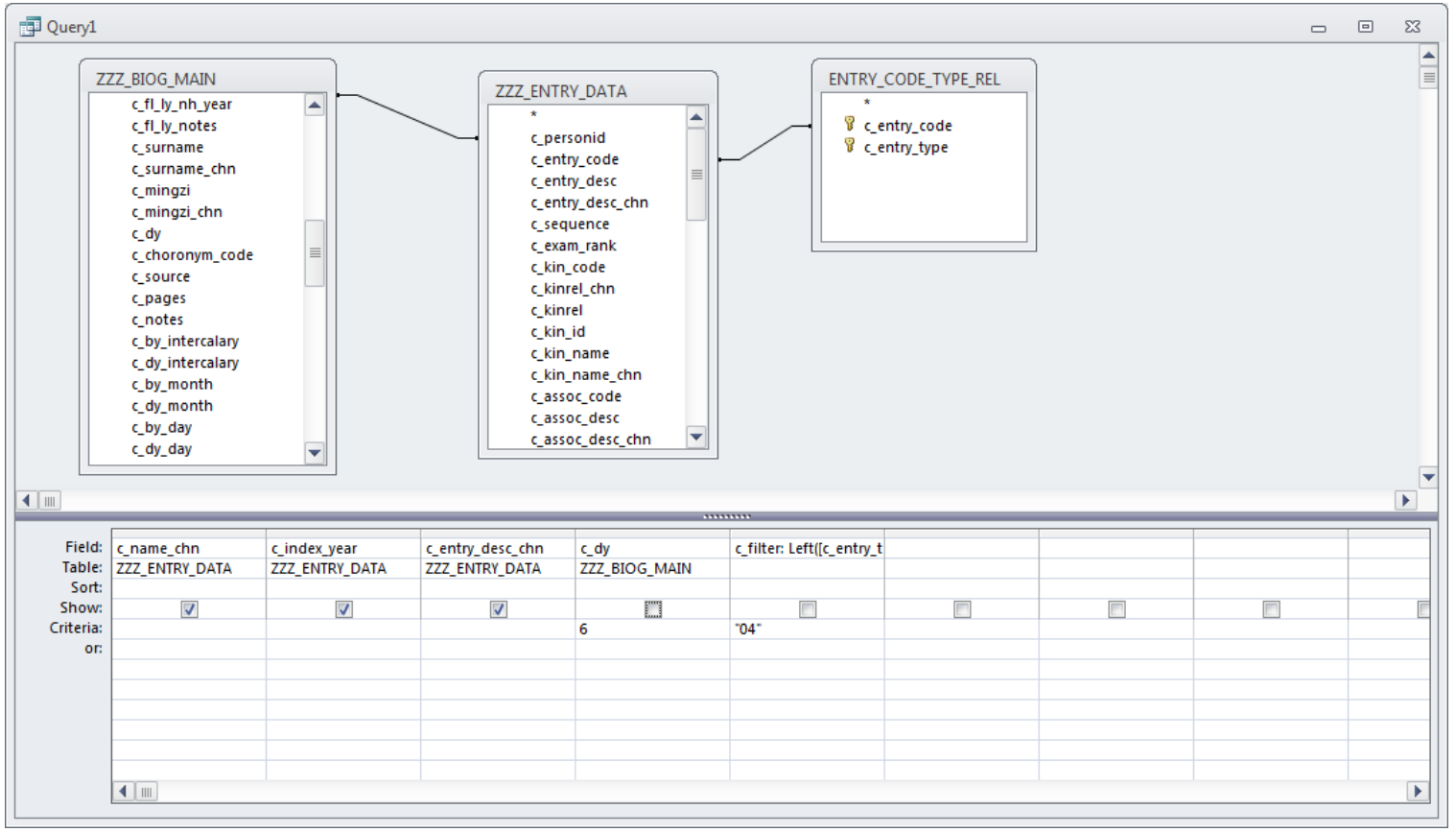
As you can see, we have 239 modes of entering service.

To simplify analytic inquiries, we have a pair of tables to aggregate these 239 modes:

ENTRY_TYPES (17 types + 7 sub-types)
ENTRY_CODE_TYPE_REL (maps codes to type/subtype)

We are now going to build a query to allow us to see all the people in the Tang dynasty who entered via examinations.

# Open the Query Builder, select the tables, and create this:




This general approach works for all the sets of tables

## Entity

**People**

Next, returning to ENTRY_DATA, note its complexity as a table (please right-click to open in Design View):

Other people often played a role in people's entry into office, and CBDB tracks both kin roles (esp. for *yin*) and social associations.

In later dynasties, specific institutions also played a role in entry into government service.

| Entity | Table |
| --- | --- |
| **Places** | ADDR_CODES |

**Places** in CBDB are specifically **administrative units**, the official designations of places within the pre-modern bureaucratic system.

The table ADDR_CODES (Please open) frankly needs work.

The goal has been to create a system of codes that match those in the CHGIS (China Historical GIS) database.

We generated our codes from Robert Hartwell's initial data to implement the rule: create a new code **only when administrative unit boundaries change**.

His data on boundaries was less than perfect, and current codes still are more bound to dynasties than we want.

| **Entity** | Table |
|---|---|
| **Places** | ADDR_CODES |

The fields in ADDR_CODES are:

| | |
|---|---|
| **c_addr_id** | the basic ID used in all other tables |
| c_name | |
| c_name_chn | |
| c_first_year | |
| c_last_year | |
| c_admin_type | this is a TEXT field, to be replaced by an ID |
| x_coord | this is still (?) the coordinates of the |
| y_coord | centroid for the polygon |
| CHGIS_PT_ID | |
| c_notes | |
| c_altnames | this violates the one-to-many rule |

| Entity | Table |
| --- | --- |
| **Places** | ADDR_CODES |

Because **Places** are administrative units, CBDB needs to track their place within the hierarchy of units, to know what units are parts of a “place” and the unit to which it reports.

This information is captured in ADDR_BELONGS_DATA. (Please open this table)

Because this table is machine-friendly / user-unfriendly, CBDB also has a second table ADDRESSES that also shows the hierarchy. (Please open.)

Each record gives an ascending list of units ending in a dynasty.

| Entity | Table |
| --- | --- |
| **Places** | ADDR_CODES |

To simplify querying, I have created a third table:

ZZZ_BELONGS_TO

It lists all the units within a given “place.”

That is, for a 州, it lists all the 縣. For a 路 or 道, it lists all the 州 **and** 縣, and so on.

| Entity | Table |
| --- | --- |
| **Offices** | OFFICE_CODES |

Representing **Office** as an entity still needs work. There are three problems:

1. Very general titles given in texts. For example, some texts simply state 參軍, but we have 153 office titles with 參軍 in them. How should we handle this?

2. Cross-dynastic analyses: at present our office codes are by dynasty.

3. Bureaucratic structure: to do good analytic studies of bureaucratic interactions, we need a good idea of who reports to whom. We have some detail for the Song dynasty and are working on the others. (It turns out that this sort of information is difficult to reconstruct.)

   Open the form frmPickOfficeTree (by double-clicking on it)

| Entity | Table |
| --- | --- |
| **Offices** | OFFICE_CODES |

In any case, the core structure of OFFICE_CODES is:

| | |
| --- | --- |
| **c_office_id** | the main ID for offices |
| c_dy | the dynasty |
| c_office_pinyin | |
| c_office_chn | |
| c_office_trans | |

No real surprises here. Since there are no foreign keys (codes that refer to other tables) except the dynasty, there is no denormalized equivalent.

| Entity | Table |
| --- | --- |
| **Offices** | OFFICE_CODES |

However, two additional tables are needed to represent the information about bureaucratic structure that we need to know. These two tables are:

OFFICE_TYPE_TREE
OFFICE_CODE_TYPE_REL

OFFICE_TYPE_TREE lists all the institutions that make up the pre-modern Chinese bureaucracy, and OFFICE_CODE_TYPE_REL links specific offices to the bureaucratic structure.

| **Entity** | Table |
|---|---|
| **Kinship** | KINSHIP_CODES |

We have far too many kinship codes (please open the table to look at them): we will need to rationalize this in the future.

The structure of KINSHIP_CODES is:

| | |
|---|---|
| **c_kincode** | the basic ID |
| c_kinrel_chn | the Chinese term |
| c_kinrel | the English term |
| c_kinpair_1 | the obverse relation for males |
| c_kinpair_2 | the obverse relation for females |
| c_upstep, c_dwnstep | kinship distance values |
| c_marstep, c_colstep | (i.e. FB = 1 up, 1 collateral) |

| Entity | Table |
| --- | --- |
| **Social Associations** | ASSOC_CODES |

As with kinship codes, we have many, many association codes (again, open the table and take a look) and no doubt will have more as we collect more data.

The structure of the ASSOC_CODES table itself is simple:

| | |
| --- | --- |
| **c_assoc_code** | the basic association ID |
| c_assoc_desc | |
| c_assoc_desc_chn | |
| | |
| c_assoc_pair | the obverse relation |
| c_assoc_pair2 | the third code in triadic relations |

| Entity | Table |
| --- | --- |
| **Social Associations** | ASSOC_CODES |

However, because we have so many specific association codes, we need to be able to aggregate them for analytic purposes. For this, we use two additional tables:

ASSOC_TYPES (please open)
ASSOC_CODE_TYPE_REL

The ASSOC_TYPES table provides categories and sub-categories under which to classify specific associations. The ASSOC_CODE_TYPE_REL table then links associations to larger categories

Let's create a query to select a type of association for Tang.

| **Entity** | Table |
| --- | --- |
| **Social Distinctiveness** | STATUS_CODES |

When we first began to develop CDBD, we started with Hartwell's categories, but as we looked at textual data, Hartwell's "status" evolved into a way of marking what people were known for in the lifetime and afterwards, hence "social distinctiveness."

The structure of STATUS_CODES is the simplest yet:

- **c_status_code**
- c_status_desc
- c_status_desc_chn

| Entity | Table |
| --- | --- |
| **Social Distinctiveness** | STATUS_CODES |

Once again, we gradually accumulated many types of social distinctions (please open STATUS_CODES) and needed to develop larger analytic categories. For this we use:

STATUS_TYPES (please open)
STATUS_CODE_TYPE_REL (please open)

These are works in progress.

| Entity | Table |
| --- | --- |
| **Social Institutions** | SOCIAL INSTITUTION_CODES |

**Social Institutions** are a new entity in CBDB, based primarily on Yuan and Ming data. At present, there are three main types: Buddhist institutions, Daoist institutions, and academies.

What should be considered a singular instance of a **Social Institution**" proves a bit complex: it can change from a Buddhist temple to Daoist and back again and still be the same institution. It can change names and even move and yet be the same institution. Our data structures capture this.

| **Entity** | Table |
| --- | --- |
| **Social Institutions** | SOCIAL INSTITUTION_CODES |

SOCIAL_INSTITUTION_CODES has the following structure:

| | |
| --- | --- |
| **c_inst_code** | a unique ID |
| **c_inst_name_code** | a name ID |
| c_inst_type_code | |
| c_inst_begin_year (etc.) | |
| c_inst_end_year (etc.) | |
| c_inst_last_known_year | |
| c_source | |
| c_pages | |
| c_notes | |

| Entity | Table |
| --- | --- |
| **Social Institutions** | SOCIAL INSTITUTION_CODES |

To complete our information about “social institutions,” however, we need two additional tables—following the rule about one-to-many relations—because an institution may have more than one name and more than one address:

SOCIAL_INSTITUTION_NAME_CODES
SOCIAL_INSTITUTION_ADDR

| Entity | Table |
| --- | --- |
| **Texts** | TEXT_CODES |

We are almost done with our survey of entities. The next to last is **Texts**.

In order to avoid “mission creep,” CBDB does not list individual titles within literary collections, nor does it record full texts nor give hyperlinks to texts (although this probably will come).

| Entity | Table |
| --- | --- |
| **Texts** | TEXT_CODES |

TEXT_CODES records core information about texts but does not list all extant editions, etc. (Please open in design mode.)

| | |
| --- | --- |
| **c_textid** | the basic text ID |
| c_title_chn | |
| c_title | |
| c_title_trans | |
| c_extant | other information |
| c_text_country | |
| c_text_dy | |
| (etc.) | |

| Entity | Table |
| --- | --- |
| **Texts** | TEXT_CODES |

The ***author*** of a text is one way in which **People** interact with **Texts**, and we will discuss the tables that record how **People** interact with other entities in the part of the presentation.

| Entity | Table |
| --- | --- |
| **Ethnicity** | ETHNICITY_TRIBE_CODES |

One can argue whether “Ethnicity” is an entity or just an attribute of **People**. I suspect that views change depending on the dynasty. It may well be an entity in the Yuan and Qing.

In any case, please open the table. At present we have 467 designations, and this is not likely to grow much.

The core structure is:

| | |
| --- | --- |
| **c_ethnicity_code** | the ID for ethnicity |
| c_name | |
| c_name_chn | |
| c_group_code | the largest category (Uyghur, etc.) |
| c_subgroup_code | |

We now have seen the main ENTITIES in CBDB. However, the heart of the database is not these entities but their interactions with **People.**

| **Entity** | Table | Interactions with **People** |
|---|---|---|
| Places | ADDR_CODES | BIOG_ADDR_DATA |
| | | POSTED_TO_ADDR_DATA |
| Offices | OFFICE_CODES | POSTING_DATA |
| | | POSTED_TO_OFFICE_DATA |
| Kinship | KINSHIP_CODES | KIN_DATA |
| Social Associations | ASSOC_CODES | ASSOC_DATA |
| | | ENTRY_DATA |
| Social Distinctiveness | STATUS_CODES | STATUS_DATA |
| Social Institutions | SOCIAL INSTITUTION_CODES | BIOG_INST_DATA |
| | | ENTRY_DATA |
| Texts | TEXT_CODES | TEXT_DATA |
| | | BIOG_SOURCE_DATA |
| Ethnicity | ETHNICITY_TRIBE_CODES | (BIOG_MAIN) |

Notes on Entities interacting with **People**

1. Because a single posting may include more than one office in more than one place, we divide postings data into three separate tables:
   POSTING_DATA
   POSTED_TO_OFFICE_DATA
   POSTED_TO_ADDR_DATA

2. Social associations can be complex, and the structure of ASSOC_DATA reflects this complexity. (Please open in Design View.)

Notes on Entities interacting with **People**

3. Please open BIOG_ADDR_CODES. These are the types of place associations we track as present.

4. Please open TEXT_ROLE_CODES. These are the types of roles for people we track. (Note that a single text may have more than one author, recipient, etc., and hence needs the separate table TEXT_DATA.)

# Part Two: Entities in the Access Query Builder

(Here we switch over to Access.)