Understanding Entity Relationship Diagrams

Introduction

Entity relationship diagrams, or ERDs as they are fondly known in the field, contain a wealth of information on the architecture of a relational database. Unfortunately, those not actively working on database design issues are likely to find them confusing, intimidating, and sometimes aggravating. In this brief tutorial I will try to dispell some of this miasma. Like so many things, it turns out that understanding a few key concepts is all that is required. This section is meant as a broad, rather high level introduction to the major concepts surrounding relational databases. This is a very broad and rapidly growing field, so that it is only possible to just barely scratch the surface. Fortunately there are now a great number of well written books on the subject.

An entity relationship diagram at best can only be considered a static picture of the "structure" of a database.

Nothing is revealed regarding the flow of data, or anything else that relates to how the data changes. However, exploring an entity relationship diagram is often a first step in understanding a databases design. Unfortunately, one gets the impression that at least in some cases the design process may have stopped with the ERD, with many incongrous columns added "just in case". To some extent, reviewing an ERD is a bit like an archeological expedition, knowning that a particular room in a ruin was used for religious ceremonies without having any idea what rituals were performed or even who might have conducted them.

There are actually a variety of possiblities used in industry to generate entity relationship diagrams. The one described here and used throughout this document is known as "Crow's Foot" notation. I believe the reason for this should become obvious shortly. Of the three approaches considered here, all used different

© NordicWrite.Se – Need help with your Nordic Academic Paper

notations, and none used "Crow's Foot". We use it here because it is the accepted *Information Engineering Methodology (IEM)* convention, it is supported by the **Visio** ® drafting program, and it is ubiquitous in articles and books describing **Oracle** ® applications. Also, once one convention is understood, it is not difficult to read diagrams using the others. For purposes of the comparisons in this document, I found it conventient to recast the three approaches into the same notational convention.

Tables

There are two basic components comprising an entity relationship diagram, entities and relationships. Entities can be invisioned as tables with columns and rows somewhat like a spreadsheet. Each column has a name and contains some certain type of information such as the latitude of a hypocenter, or the arrival time of some seismic phase. The table itself also has a name by which it is referenced. Rows represent a particular instance of what the table represents. For example, an origin table might contain one row for each location that has been calculated. In the old days, columns were referred to as attributes, and rows were called tuples. It may be helpful to think of a relational database as a set of related spread sheets of varying dimensionality. On the other hand, maybe it won't.

In general a database contains only one or at most a few tables containing the same kinds of data. For example, one would expect to find a table containing one row for each entry in a local earthquake catalog. You wouldn't find a separate table for different regions or periods of time (months or years). Instead, tables contain all of the data of a particular kind and columns designated as keys uniquely identify a given row. Frequently a key will be represented by a single column, although sometimes a few columns might be required. As an example, consider the local Id used in the CUSP system. For a local database, this number is sufficient to uniquely identifiy any earthquake in the database. However, a database containing hypocenters from multiple networks might require two columns, one identifying the source network in

© NordicWrite.Se – Need help with your Nordic Academic Paper

addition to the CUSP Id. Sometimes, although not frequently, there are two or more sets of columns that independantly identify the rows in a table and where either could serve as keys. Of these, one set is designated the "primary key", and all others are termed "alternate keys". There is one other kind of key called a "foreign key". Although these are generally not unique within a given table, they do uniquely identify the row of some other table or collection of tables. They are, in fact, identical to the primary keys in these tables. Foreign keys are used to form "relationships" between the table that contains them, and the table for which they are the primary key. For the earthquake catalog example, there might be a column designated as a region code, whose values would be primary keys in some "region information" table containing one row for each possible region. There are generally several columns in a table that are not part of any keys. These are simply data columns, and might contain such information as an earthquakes magnitude estimate, or the number of phases used to locate it.

Relationships

The lines with symbolized ends connecting database entities represent the relationship between two tables. Each line describe both directions of the relationship. There are four basic relationships that can be used to describe the relationship of one table to another. The diagram on the right shows these four possibilities by discussing the relationship of some entity **B** with another entity **A**. The first relationship shown is that of a one to one relationship between the two tables. Specifically this means that for every row in table **A** there is one and only one corresponding row in table **B**. This requires that the number of rows in **A** must be the same as the number of rows in **B**. Such relationships represent a kind of a

© NordicWrite.Se – Need help with your Nordic Academic Paper

choice, in that it would be possible to merge two tables related in this manner into a single table sharing all of the columns of both. Alternatively, and particularly for seismic applications, it might be better to keep two tables distinct, with **B** containing information of a specialized and possibly site specific nature, while **A** might contain
information of a much more general nature. Generally one to one relationships are uncommon in database design.

KEY: Key to ERD relationships.

The second relationship shown is much more common, and will help to understand the bidirectional nature of this notation. Simply put, the symbol is to be interpreted that every row of **B** is uniquely related to specific row of **A**, while every row of **A** is related to at most one row of **B**. As a pneumonic, I sometimes think of the symbol near the **B** entity as reading "0 or 1", since that is what it looks like. One thing that is difficult to keep sorted out, speaking for myself, is the directionality of the relationship. One can think of the terminal symbol as an arrow head, defining how one entity, **A** in the example, is related to another, **B**. Apparently some of the rows in **A** do not relate to any rows in **B**, and the corresponding "foreign key" is said to be null. On the other hand, every row of **B** is related to exactly one row of **A**. One can see, that table **A** must contain at least as many rows as table **B**, and possibly considerably more.

© NordicWrite.Se – Need help with your Nordic Academic Paper

With the third relationship shown in diagam **KEY**, things get quite a bit more interesting. This notation reads that a given row of **A** is related to one or more rows in **B**. To accomplish this, particular values of some foreign key in **B**, which is also the primary key of **A**, might be duplicated one or more times. For example, a column in a table holding information on the arrival times of particular seismic phases might have a column (foreign key) containing an ID, which is the primary key of a table of hypocenters. In this example, the phase table is **B**, and the hypocenter list is **A**. In this way multiple phase arrivals can be "assocated" with a given hypocenter. The simple example below, however, shows that the situation is generally a bit more tricky than this for real applications. Note that every row of **A** must be associated with at least one row of **B**. The situation where a row of **A** corresponds to no row in **B** is shown in the fourth symbol. This reads that rows of **A** are associated with zero or more rows of **B**. For the phase arrival example, this would mean that there could be a location for which no phase data was available. The location, perhaps, was simply typed in without supporting data.

So far the discussion has been limitted to the symbol on the right side of the connector. In each example, the left hand symbol is shown as a unique connection. In fact the situation is a bit more tense than this, so that symbols can occur on each end in various combinations. For the most part this is not a concern, as the relationships in alternating directions can be considered independant. The exception to this is that of having any kind of "crow's feet" on both ends. The problem is that such relationships cannot be represented in a normalized database, and thus cannot occur without introducing an additional linking table. This leaves 12 permittable permutations, or only 7 if reflexive pairs are taken as one. How this works out in a realistic example is shown in the following section.

A Simple Seismic Example

© NordicWrite.Se – Need help with your Nordic Academic Paper

For the purposes of this tutorial I have chosen a familiar set of four entities directly related to the storage of earthquake data and summary information. This diagram demonstrates all of the important concepts required to understand the discussions in other sections of this document. The titles across the top show how I have classified the basic entities into four categories, **DATA, LINK, SUMMARY, and CATALOG**. From left to right, these categories reflect increasing degrees of processing and analysis. The **DATA** category contains information directly related to observations, for example pointers to waveform data, or tables of parameters directly extracted from waveforms, such as arrival times and amplitudes. The **SUMMARY** category contains tables that generalize information from the **DATA** category such as earthquake hypocenters and magnitude estimates. For example an earthquake hypocenter requires at least 4 arrival times from the **DATA** layer. Which arrival times are associated to a particular hypocenter, for example, is contained in the **LINK** layer. Finally, on the far right, the **CATALOG** layer contains the general index into the database. For seismic network applications, this is typically an earthquake catalog containing one entry for each significant event. Generally, especially immediately following a significant event, several trial locations might exist, with only one eventually evolving into the definitive solution.

In each of the four entities in diagram **ERD** below the name of the table appears in the upper rectangle, and key column names are shown in the lower rectangle. Primary keys are underlined.

ERD: Fragment of the IDC operational schema showing the tables related to arrivals (picks), locations, and a summary catalog.

© NordicWrite.Se – Need help with your Nordic Academic Paper

The two entities on the right are present in all three of the seismic schemata described in this report. The basic idea is that there is an "origin" table and an "event" table. The "event" table can be thought of as the "gold standard" earthquake catalog, in that it has one and only one entry for each cataloged significant event. Its primary key, "evid", is a simple unique serial number with no other information content. The "origin" table is similar, but contains possibly multiple locations for each event. This table has a primary key, "orid", which is also a vanilla identifier. Such could arise in a network that has both automated and manual processing, or when combining locations from adjacent regional seismic networks. Depending upon some factor such as the region of occurrence, one or another of these would be determined to be the "correct" location. This would be the preferred origin referenced in the "event" table by setting its "prefor" foreign key to the "orid" of the appropriate entry in the "origin" table. This relationship is carried by the connector labelled "prefor" in the figure. All rows in the "event" table must reference an entry in the origin table through the "prefor" foreign key. Each row in the "origin" table is referenced by no more than one row in the "event" table via its "prefor" foreign key. The upper connector reflects the contents of the "evid" foreign key in the "origin" table. When the column names of both the foreign keys and the primary keys they reference are the same, the relations are unnamed to prevent clutter. The open circle and bar on the right end of this connector indicates that there can be rows in the "origin" table that are not represented at all in the "event" table. These would have null values in their "evid" column, and might represent locations that are marginal, or possibly outside the region of interested and therefore not included in the standard catalog. The crow's foot with bar on the left indicates that each event in the "event" file must be associated with at least one and possibly many rows in the "origin" file. These are indicated by setting "evid" in the "origin" table to the value of the "evid" primary key in the "event" table. This is typical of the manner inwhich the one to many relationship arises in relational databases. Note that a row cannot exist in the "event" table unless at least one row in the "origin" table references it.

© NordicWrite.Se – Need help with your Nordic Academic Paper

Maintaining this condition during updates is within the purview of the processing application and are known generally as "business rules".

The relationship between phase arrival and calculated origins is expressed by the three relations on the left.

The "assoc" table is the critical link between rows in the "origin" table and the phase arrival data in the "arrival" table that supports it. Such a table is required whenever a many to many relationship must be expressed. Each row of the "assoc" table contains exactly two columns, which together are unique and can be taken as a primary key. Each pair can also be thought of as two foreign keys, "arid" connects the "arrival" table, and "orid" connects to a particular row of the "origin" table. Those rows of the "assoc" table with a common "orid" represent all of the data supporting a particular location in the "origin" table. Similarly, those rows in the "assoc" table with a common value of "arid" represents the list of all locations that a particular phase arrival was used as a datum. Similar link tables are used to associated amplitudes with magnitudes and so forth. Notice that the the connectors on either side of the "assoc" table have different symbols. The one or many notation on the left (bar with a crow's foot) shows that all locations require at least one arrival. The none or many symbol on the right (circle with crow's foot) indicates that some phase arrivals might not be associated with any locations. They might arise in a real-time transaction system when a noise burst is misidentified as a seismic arrival, or perhaps they are true arrivals for an event that is too small to provide enough arrivals to support a location. In either case, such orphans could eventually be expunged from the database.

I hope you found this discussion of a simple relational approach to a familiar problem to be helpful. Frankly, I find the whole process of learning about something I knew almost nothing about to be extremely terrifying.

I found the book "Oracle X.X for Dummies" to be a rather good starting point, although a desire for "chewier" fare rapidly ensued. For years I was put off by this series because of the implications with

© NordicWrite.Se – Need help with your Nordic Academic Paper

respect to the readership. I think now that this may have been a mistake, and besides, book covers are cheap.

© NordicWrite.Se – Need help with your Nordic Academic Paper

References

1. Coordinators, N. R. (2018). Database resources of the national center for biotechnology information. Nucleic acids research, 46(Database issue), D8.
2. Larson, B., Shapiro, R. A., Stanfill, C. W., and Weiss, A. H. (2018). U.S. Patent No. 9,977,659. Washington, DC: U.S. Patent and Trademark Office.
3. Cleaver, M. J., and Boone, R. A. J. (2019). U.S. Patent Application No. 10/318,524.
4. Larson, B., Shapiro, R. A., Stanfill, C. W., and Weiss, A. H. (2018). U.S. Patent No. 9,977,659. Washington, DC: U.S. Patent and Trademark Office.

© NordicWrite.Se – Need help with your Nordic Academic Paper