Databases

Data and Information

While data is known, discrete facts that have been stored and recorded, information is data placed in context and presented. It is much more useful to humans. SQL is a tool for extracting information from a database.

Metadata

Metadata is *data about data*. For example type, length or description. It helps us to keep data storage consistent, useful and meaningful.

A database is a large, integrated, structured collection of data, used to model some real world enterprise as entities and relationships. A Database Management System or DBMS is used to interface with a database. Databases differ from simple programs interfacing with files by avoiding redundancy and ensuring consistency. They also allow better file sharing and can improve development speed and reduce maintenance.

Database Development Process

- Database Planning
- Systems Definition
  - Enterprise data model, where the components and interactions of a business is defined.
  - Specification of scope and boundaries of the system.
- Requirements Definition and Analysis
  - Take in requirements for the system and analyse them to define a system that will satisfy them.
- Design

    - Conceptual Design - construction of model of the data to be held in the database, independent of any technical considerations. Generally using *entity relationship* (ER) diagrams.
    - Logical Design - technical decisions for the conceptual design above. While in this subject this will always be a DBMS, it could also be a JSON document or even a spreadsheet.
    - Physical Design - implementation details of a given logical design; relations, data types, configurations, etc. Specification of types can help to make a database smaller and faster. It's important to consider all the factors that inform a datatype.

- Application Design
    - In parallel with the design phase, design of the application continues.
- Implementation
- Data Conversion and Loading
- Testing
- Operational Maintenance

Entity-Relationship Models

An entity-relationship model is a method for modelling data needed for an organisation. An entity is a real-world object, distinguishable from other objects. An entity set is a collection of similar entities. For example an entity might be "John Smith", of the entity set person. John might have attributes like name, date of birth, etc. Each entity has a *key*. This is a unique piece of data describing the record.

In a visual model, an entity is represented as a rectangle with attributes in ovals connected to it by lines. The name of the key is underlined.

A relationship is an association between entities. For example, John might *live in* Melbourne, a City. In this case, City is another entity set with its own attributes. The relationship might have an attribute like "since". A relationship is depicted as a diamond, connecting to the relevant rectangles, with attributes as ovals.

A relationship might connect to the same entity twice, such as in a "reports to" relationship between two employees.

Constraints

Relationships have constraints on them, describing which entities can be related by them. On of these is key constraints, which are

- Many-to-many. Any number of entities on one side are related to any number of entities on the other side. For example a lecturer may teach many lectures, which may be lectured by many lecturers.
- One-to-many. A given entity may be related to a large number of another entity.
- One-to-one. A given entity is related to a single other entity. For example a person is part of one department.

A key constraint indicating "many" is simply a plain straight line. In a "one" relationship, known as a key relationship, an arrow is drawn from the side which has only one incidence to the relationship. If every entity in a set has a given relationship, we bold the line describing that relationship.

A weak entity is one which can be uniquely identified only with knowledge of another owner identity. To indicate this, we use a identifying relationship. All weak entities must have this relationship to their owning identity. To show this on a diagram, we bold the weak entity, the line (which will always be arrowed, as each weak entity has one owner) and the relationship diamond and underline the key of the weak entity to indicate it doesn't uniquely identify the entity.

Some special attribute types exist. These include

- A multi-valued attribute, where multiple different pieces of information of the same type are stored. For example, all of the phone numbers associated with a person. This is depicted with two concentric ovals rather than a single oval.
- A composite attribute, which is just an attribute with sub-attributes. Shown by given the attributes attributes in the same way one would entities.

Data Modelling

A data model allows the translation of real world concepts into computational structures. One form of this is relational modelling, as in a relational data base. A relationship between tables in a relational database takes the form of a foreign or primary key.

A relational database is defined by a schema, which is a specification of relations between tables and name and type of each column in a table.

The *cardinality* of a table refers to its number of records. The degree or *arity* of a table refers to its number of rows.

ER to Relational Models

An entity in an entity-relationship model becomes a table in a relational database. This is the transition from conceptual design to logical design. To progress from this stage to physical design, we choose datatypes for the columns. The implementation stage is then undertaken by creating the related SQL for this physical design.

To implement our relationships and constraints we need to use keys. A key is a form of integrity restraint. A primary key enforces uniqueness in a table. A foreign key ensures that only existing records can be referenced from other tables. Some important definitions exist for keys.

- A set of fields is a *superkey* if no two distinct tuples can have the same values in all key fields.
- A set of fields is a *key* if it is a superkey and no subset of the set is a superkey.
- Of all keys, one is chosen as the *primary key*.
- Non-chosen possible primary keys are *candidate keys*.

A foreign key must always point to another table's primary key. A database has referential integrity if all foreign keys are enforced.

An integrity constraint is a condition that is applied to a column, such as a domain constraint for a column. A legal instance of a relation (table) satisfies all such constraints.

For multi-valued attributes, we often need to split them up into multiple attributes such as a home and work phone for phone numbers. Likewise, for composite attributes, we simply flatten them and include all columns in the relevant table.

To create relationships we often create new tables where we take a primary key from each of the involved tables and add any relevant data about the relationship as additional columns. In instances where we have a "one" relation it is generally preferred to include this key in the table with the singleton relationship; succinctly "the primary key from the many side becomes a foreign key on the one side". If the key is mandatory, we simply mark it as `NOT NULL` in our SQL.

For a weak entity which exists only in its relation to another entity, we implement cascading deletion.

```
CREATE TABLE dependent (
    name CHAR(20) NOT NULL,
    age INTEGER NULL,
    cost DECIMAL(7,2) NOT NULL,
    employee_id CHAR(11) NOT NULL,
    PRIMARY KEY (employee_id, name),
    FOREIGN KEY (employee_id) REFERENCES employees ON DELETE
        CASCADE
);
```

In the above example we have a table storing employee insurance dependents, which are meaningless without their related employees.