Relational Data Model

Definitions

- An entity class is a collection of entity instances that have a common structure. For example, a whole collection of student records could form an entity class.
- An entity instance represents a particular object of interest that is to be represented and tracked. For example, a student record is an entity instance that represents an individual student.
- An attribute represents a piece of interesting information, or a measurable fact, about the instances of an entitity class. For example, year of first registration is a fact about students that might be represented as an attribute of all the instances of the entity class student
- A domain is a set of values that can be assigned to an attribute; for example, the attribute birthday coudl be given values from the domain date.
- A relationship is an association between entities. Entities are often identified by nouns in a requirements specification, and relationships by verbs. For example, owns might form a relationship between entities person and vehicle. Relationships can be described as relationships between entity classes, or between entity instances.
- Mathematically, a relation consists of a heading, which is a subset of the Cartesian product of a set of (attribute name, domain) pairs, and a body, which contains (attribute name, value) pairs. For example, the entity class student could be represented as a heading, (student number, integer), (student name, text) and a body containing values like (student number, 123),(student name, Bloggs) A relation is implemented as a table in a relational database.
- A candidate key is a minimal set of attributes that identifies each individual row in a table (each tuple in a relation). For example, suppose there was a relation Slotroom,day,time in a timetabling application. Then room,day,time or class,day,time would serve as alternative candidate keys for the relation.
- The primary key is the candidate key that has been nominated to identify individual rows in a table. For example, in the timetabling relation above, room,day,time would be likely to form a suitable primary key because class is likely to change.

Database Integrity

ACID

- Atomicity - something is either done completely, or not done at all. The state of doing it is not visible outside the database.
- Consistency - The database is in a legal state at all times. When a transaction occurs, it can not break the rules. These rules are about integrity, what is allowed and what is not allowed in certain locations of the database.
- Isolation - There can be more than one transaction occurring at the same time. A certain transaction will not see changes made by other transactions.
- Durability - When a transaction is done, it will be committed. After it is committed, it can no longer be undone.

Definitions

- Enterprise rule: empirical constraint on real world entities and attributes. Examples: "Each pallet contains 5184 bottles" (a real example, referring to supplies of an antiviral); a student is normally allowed at most two attempts at a module examination.
- Data integrity: the data in a database models the real world; the database corresponds to reality. For example, a person has exactly one date of birth.
- Integrity constraint: a constraint on the values or combinations of values that are allowed to be entered into a database. For example, 'no two distinct vehicles are allowed to have the same vehicle identication number' is a constraint that allows 'vehicle identication number' to identify a particular vehicle record.
- A domain is the set of possible values for an attribute; it is the type of the attribute. In a relational database, a domain is the set of possible values for the cells in a column of a table.
- A candidate key is a collection of attributes whose combined values are different for each tuple in a relation. A candidate key is also minimal in the sense that no subset of the candidate key will identify tuples in this way.
- A foreign key is a collection of attributes from one relation that constitutes a candidate key for another relation. The values of the foreign key attributes in the first relation must also be present in some tuple of the second relation.

Constraints

- Other kinds of constraint are needed because domains and key constraints are not sufficient to capture all the different kinds of enterprise rule that need to be modelled in a database.
- For example, the constraint every sheep farmer owns at least one sheep cannot be represented using domains, foreign keys and candidate keys.
- In general, minimal cardinality constraints (1.. cardinalities) demand more than domains and foreign and candidate keys.

SQL Integrity Checks

Enums:

```
CREATE TYPE character_kind AS ENUM
  (monster, wizard, hero, seer);
  CREATE TABLE character (
  ...,
  kind character_kind,
  etc
);
```

Check contraint:

```
CREATE TABLE character (
  ...,
  kind text CHECK
    (kind in (monster, wizard, hero, seer)),
  etc
);
```

Foreign Key:

```
CREATE TABLE character_kind ( kind text primary key );
  INSERT INTO character_kind (kind) VALUES
    (monster), (wizard), (hero), (seer);
  CREATE TABLE character (
  ...,
  kind text REFERENCES character_kind(kind)
);
```

Merits of these approaches:

- Enumerated type and foreign key reference allow constraint to be implemented once, and reused in many tables.
- Both these approaches allow a simple query to display values to be entered into the table via form widget.
- However, acceptable values are not immediately visible in the table definition when either of these approaches is used.
- Cannot use native string operators such as like with enumerated types (this restriction is not true of every DBMS.)
- Using a foreign key reference facilitates modification of the list of acceptable values.
- A check constraint is immediately visible in a table definition.
- Values shown in the check constraint can be used with native string operators.
- A check constraint is not available for use in other tables.

NoSQL

NoSQL Servers

- MongoDB: BSON, binary format JSON
- MarkLogic: XML with support for JSON and other formats
- Apache CouchDB: JSON
- Apache Cassandra: key-value store

Advantages

- Scaling using clusters of commodity hardware rather than bigger specialist servers
- Capacity to handle larger volumes of data and higher transaction rates than rdbms
- Less need for database administrators
- Lower costs, both to start up and to expand
- Few, if any data model restrictions

Disadvantages

- NoSQL databases are relatively immature, so expert support can be difficult to obtain
- NoSQL data models emphasise whole documents, which eliminates the need for JOINs but which also makes analysis of data sets difficult
- There are many NoSQL data manipulation languages, which reduces portability of queries and transferability of skills
- The lack of a schema is likely to present problems for maintenance as a database matures
- Because data is not normalized, maintaining consistency is challenging

Map reduce

- A MapReduce job splits input data, in the form of (key,value) pairs, into chunks that are processed in parallel.
- A MapReduce job configuration typically specifies mapping, combination, partitioning, reducing, and input and output formats. A MapReduce job configuration to count the occurrences of words in text files distributed across a network could use the original files as chunks, and specify a map operation to count words, and a reduce operation to combine word counts.
- A map task operates on a single chunk of data, producing as output a collection of (key,value) pairs. A map task in the word counting application could count the words in a single text file, producing a list of words each paired with its number of occurrences.
- A reduce task takes two or more collections of key,value pairs and reduces these to a single collection. A reduce task in the word counting application might take several lists of words with their respective word counts and deliver a single list that gave the total count for each word.

Eventual Consistency

- Eventual consistency means that if a data item is not written or updated for a sufficiently long period of time, then all reads of that item will return the same value.
- An application developer must allow for the possibility that an application may read data that has been superseded.
- Eventual consistency makes it easier to provide readily scalable, highly available distributed systems that continue to operate even when nodes or connections between nodes fail.

Semistructured Data + XML

FLOWR

FOR, LET, WHERE, ORDER BY, RETURN.

Security