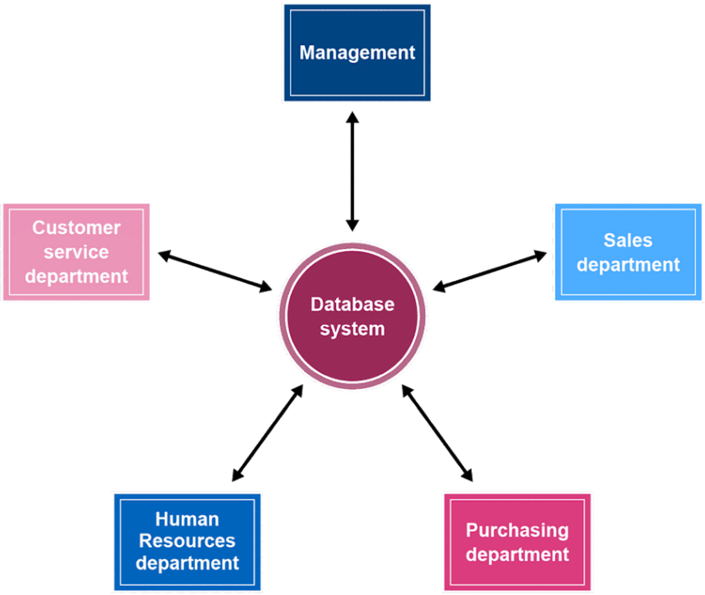
Databases

University of Essex

Databases were first thought of as a way to reduce redundancy on records and data, if different users with different needs use data, it is possible that in the long run data may have redundancies or discrepancies. This is solved following the database model, where different users have access to the same data.

In this way, the redundancy of data is reduced, since repetitions are avoided. We also reduce the possibility of inconsistencies. In addition, it guarantees shared access to data by many users operating simultaneously.



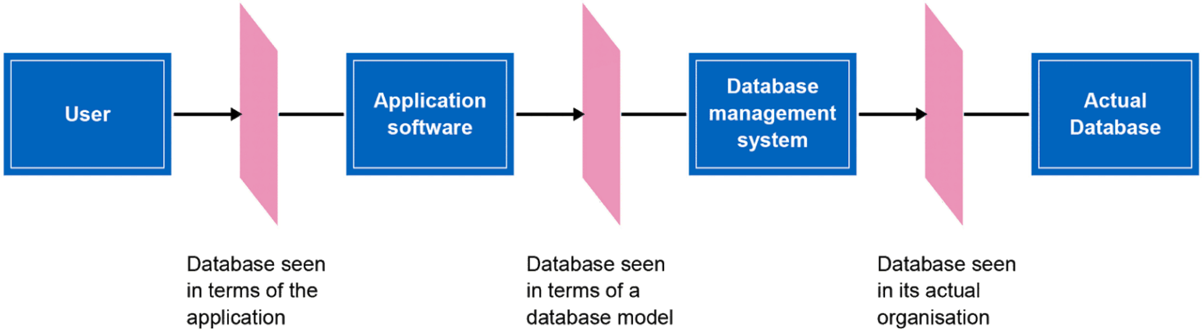
b. Database-oriented information system

Database management system

A Database Management System (DBMS) is a software system designed to manage collections of data that are large, shared, and persistent.

Persistent data, such as financial records, refers to data that contains core information, is unlikely to change and is stored on a hard drive or disk. Some amendments/updates may be needed but the information stored is relatively permanent, unlike transient data which is discarded at the end of a session.

Database Model


User
Application software
Database management system
Actual Database
Database seen in terms of the application
Database seen in terms of a database model
Database seen in its actual organisation

A conceptual view of a database. There are various types but our focus here is on the Relational Data Model.

This represents a database in the form of relations; a 2-dimensional table with rows and columns of data (see below). A database may contain one or more such tables. A relation schema is used to describe a relation.

E.g. STUDENT(Name, SSN, HomePhone, Address, OfficePhone, Age, GPA)

A Relation with a Key

The Role of Schemas

A Conceptual schema describes the real-world events to the users of the database system about the data required.

- Schema - A description of the structure of an entire database. Used by database software to maintain the database.

- Subschema - A description of a portion of the database pertinent to a particular user's needs. Used to prevent sensitive data from being accessed by unauthorised personnel.

A Key (K) of a Relation (R) is a subset of the attributes of R which has the following time-independent properties:

- Unique identification: The value of K must uniquely identify each tuple in R; and
- Non-redundancy: Other non-key attributes must maintain functional dependencies with the value of K.

Primary Key

There may be more than one set of attributes which satisfy the time-independent properties of a key. Each of them is called a candidate key. In this case, one of the candidate keys must be chosen as the primary key.

Primary key satisfies the following properties:

- The attribute(s) uniquely identifies each tuple in the relation (uniqueness); and
- The key cannot be null

Foreign Key

We relate information stored in different tables with the help of a common attribute.

Foreign Key (Referential Integrity): A set of attributes is a FK of relation schema R1 if, and only if, it is the primary key of another relation schema R2.

Relational Design Issues

Insertion

When inserting a new tuple, we need to ensure that the database constraints are not violated:

- The value of an attribute should be of the correct data type, i.e. from the appropriate domain;
- The value of a prime attribute, i.e. the key attribute, must not be null;

- The key value(s) must not be the same as that of an existing tuple in the same relation; and
- The value of a foreign key, if any, must refer to an existing tuple in the corresponding relation.

If the constraints are violated, there are two options available to the users:

- Reject the operation; or
- Rectify the operation.

Deletion

To remove an existing tuple (t) from a relation (R). When deleting a tuple, the following constraints must not be violated:

- The tuple must already exist in the database; and
- The referential integrity constraint is not violated.

If the constraints are violated, there are four options available to the users:

- Reject the operation;
- Rectify the operation to find the existing tuple concerned;
- Also remove the tuples that reference the tuple being deleted; and
- Modify the referencing attribute values, i.e. values of foreign keys, that cause the violation.

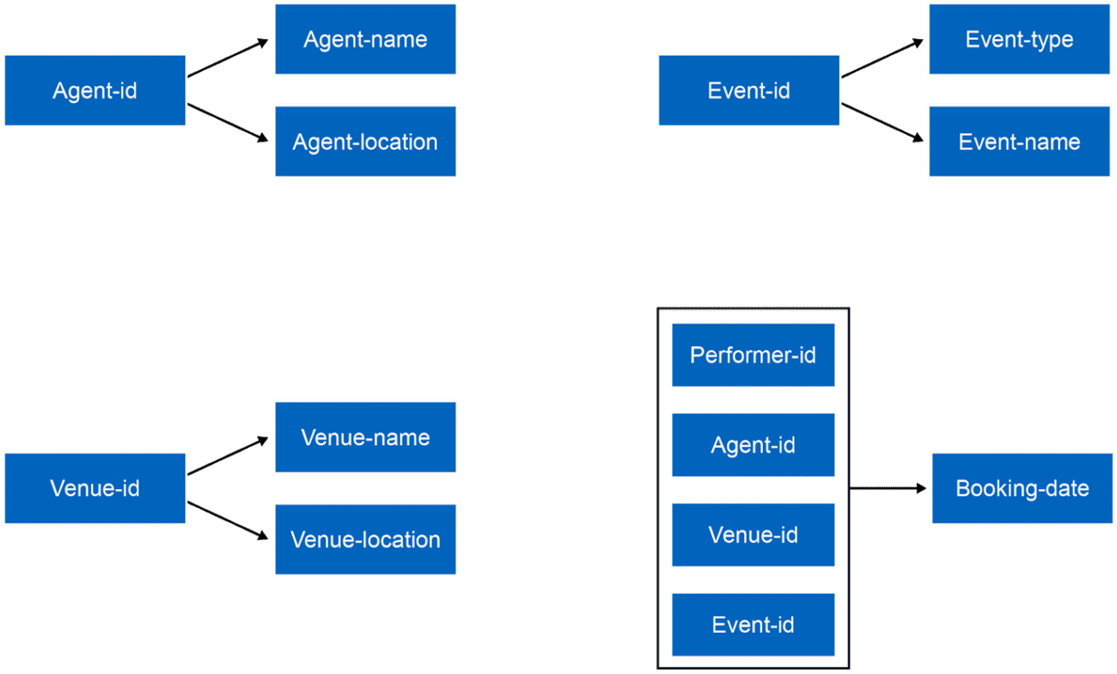
Good Practices in Relational Design

In order to ensure that the data is “good” the process of normalisation is applied.

- No redundant information in tuples;
- No update anomalies: modification, deletion, insertion; and
- No NULL values in tuples.

Normalisation - Dividing the columns of a relation into two or more relations, duplicating those columns necessary to maintain relationships.

Lossless decomposition - An approach for a fully normalised database. A “correct” decomposition that does not lose any information.



Brookshear, J. (2018)

To **avoid data redundancy problems**, a database needs to go through a Normalisation process, i.e. 1st, 2nd and 3rd or even Boyce-Codd Normal Form.

Structured Query Language (SQL)

Structured Query Language (SQL) is a database language used to manage, manipulate and retrieve data from a database. SQL syntax is considered to be a user-friendly language.

Basic SELECT-FROM-WHERE Construct

The Data Manipulation Language (DML) component of SQL allows for the retrieval, insertion and update of the data in a database. It is sometimes called the "query" language. The basic SQL query statement takes the so-called S-F-W form

- SELECT <attribute list>: Where <attribute list> is a list of attribute names whose values are to be retrieved by the query.
- FROM <table list>: Where <table list> is a list of the relation names from which the attributes to be retrieved.
- WHERE <condition>: Where <condition> is a Boolean expression used to specify the condition that the tuples to be retrieved have to satisfy.

If the WHERE clause is not specified, no condition is imposed on the selection of tuples. As a result, all tuples of the relation specified in the FROM clause are selected.