The DBMS
A database system consists of to parts; the DataBase Management System, which is the program that organizes and maintains the lists of information, and the database application, a program that lets us retrieve view or update the information in the lists. In short the DBMS provides the following services:

- Data definition; provides a method of defining and storing a data population.
- Data maintenance; maintains the population using a record for each item in the population, with fields containing particular information that describes that item.
- Data manipulation; provides services that let the user insert update, delete and sort the data in the database.
- Data display; (optional) provides some method of displaying the data.
- Data integrity; provides methods of ensuring that the data is accurate.

Transaction Processing and ACID test
I append a section from Philip Greenspun's book (http://www.arsdigita.com/books/panda/databases-choosing)
Data processing folks like to talk about the "ACID test" when deciding whether or not a database management system is adequate for handling transactions. An adequate system has the following properties:

- ***Atomicity:*** Results of a transaction's execution are either all committed or all rolled back. All changes take effect, or none do. That means, for Joe User's money transfer, that both his savings and checking balances are adjusted or neither are.
- ***Consistency:*** The database is transformed from one valid state to another valid state. This defines a transaction as legal only if it obeys user-defined integrity constraints. Illegal transactions aren't allowed and, if an integrity constraint can't be satisfied then the transaction is rolled back. For example, suppose that you define a rule that, after a transfer of more than $10,000 out of the country, a row is added to an audit table so that you can prepare a legally required report for the IRS. Perhaps for performance reasons that audit table is stored on a separate disk from the rest of the database. If the audit table's disk is off-line and can't be written, the transaction is aborted.
- ***Isolation:*** The results of a transaction are invisible to other transactions until the transaction is complete. For example, if you are running an accounting report at the same time that Joe is transferring money, the accounting report program will either see the balances before Joe transferred the money or after, but never the intermediate state where checking has been credited but savings not yet debited.
- ***Durability:*** Once committed (completed), the results of a transaction are permanent and survive future system and media failures. If the airline reservation system computer gives you seat 22A and crashes a millisecond later, it won't have forgotten that you are sitting in 22A and also give it to someone else. Furthermore, if a programmer spills coffee into a disk drive, it will be possible to install a new disk and recover the transactions up to the coffee spill, showing that you had seat 22A.

DBMS models
The DBMS availible today can be grouped into five different types, or as they’re commonly called: DBMS models. There are; File Management System, Hierarchical Database System, Network Database System, Relational Database Model and Object-Oriented Database System. The FMS and the HDS we can forget, they will never be able to match or needs. The NDS looks promising but will fail in our case due to complexity of the problem. The OODS has no standard yet, and each vendor has his own description for what it is and how to implement it. Leaving us the RDM.

Information and data
A few fundamental concepts relate information and data to the computerized storage of information using database software. The concepts use 3 “realms” in talking about data stored in an information processing system; the realm of the real world, the realm of ideas and the realm of data.

- *The realm of the real world.* The first realm is that of the real world in which entities exist with certain properties. An entity is something about which information is stored. An intity can be tangible or intangible. All entities have properties about which information can be stored (entity color, size, value, …). An important task in constructing a database is to identify real-world entities about which information is to be stored and there properties.
- *The realm of ideas.* The second realm is that of ideas, where information is used to symbolically represent entities and their properties. In this realm, knowing how the information is stored is not

important (on which media). The unit that stores information about a single entity is called a record. A piece of information that represents a single property of an entity is called an attribute.
- *The realm of data*. The third information realm is that of data, where strings of characters or bits are used to encode information to be stored in a computer system. Many terms are used in this realm. The most imported is called data element, which refers to a single, atomic piece of data that can not be subdivided and still retain any meaning. Data are stored in fields.

Enhancing security
The centralized storage of data in a database, and the accessing of this data by multiple end users and application programs, bring with them the need for security. There must be mechanisms that will allow users to access the data they need but will prevent them from accessing data they are not authorized to see. In addition to controlling the data particular user has access to, the database software may also control the type of access the user has (read, write, delete, add …). To provide for security facilities, many database software packages provide a way of authorization.

Database personnel roles
- *Data Administrator*. In order to effectively control the database environment, there must be one key individual who is responsible for the overall centralized control of the data. The job of the data administrator is not a technical one. It concerns policy-oriented tasks regarding data strategy and overall data planning.
- *Database Administrator.* His task is more technical. He has the overall control over the DBMS that the organization employs and may perform the technical tasks associated with installing and maintaining it.
- *Database Designer.* This role is often performed by members how are closer to the end user than the data administrator and the database administrator. However they will work closely with the data administrator and the database administrator to ensure that the design work is consistent with overall concept. They must perform two very different tasks; logical database design and physical database implementation.
- *Application Developer*. Application developers analyze and document the needs of end users and then develop the application program interface(s) (API) that will access the database.

Data definitions and views
When talking about the data stored in the database it is common to look at the data in at least three different ways. They are:
- *User view.* Is a set of data and relations between them as they are perceived by one or more users or API's.
- *Logical data model.* This is the entire collection of data elements, and the relationships between them. It indentifies all the different types of data stored in the database. It combines all the individual user views into one large, composite structure. The logical data model does not address how the data is stored physically, nor does it address the requirements of the DBMS. To develop a logical data model the database engineer collects the data required to implement the various user views and documents the relationships that exits between them. Here we try to reduce complexity.
- *Physical data model.* The actual files, data elements, and indices that implement the database on the physical storage medium. In deciding on specific physical implementations, the designer must take into account the requirements of each individual application (access time, # of records to update …). Physical organization has to deal with such issues as efficient use of storage, speed of access, addressing techniques, keys, indices, restart and recovery requirements, frequency of use …. Complexity can not be avoided and can be advantageous in the physical organization.

ANSI/SPARC Architecture
The American National Standards Institute formed a study group to investigate DBMS. This group was called the System Planning and Requirements Committee. The defined three data definitions or views:
- *External schema*. A external schema corresponds directly to an user view.
- *Conceptual schema*. The conceptual schema is a computerized representation of a logical data model.
- *Internal schema*. Is the computerized representation of a physical data model.

Entity-relationship diagram
Entity-relationship diagramming is an important technique for analysis and diagramming. An entity-relationship diagram is a chart that shows which entities are important and what relationships exits

among them. Entity-relationship diagrams serve as useful tools on which more detailed data models can be based. The entities are place in square boxes (1entity/box) lines showing the relationships.
We have tree relationships or associations.

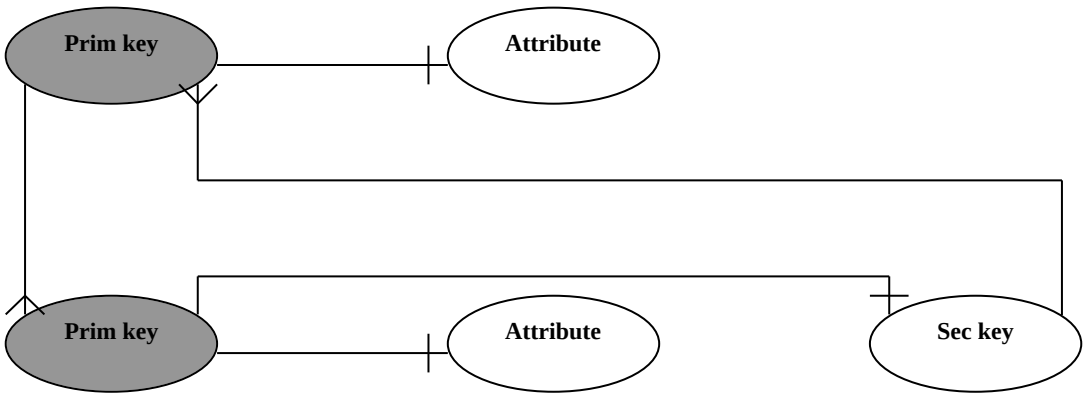
- *One to one*. We use ------|- to show a one to one, -|-----|- in both direction and ------0|- to represent a one to zero ore one.
- *One to many*. We use ------< to show a one to many, -----|< one to at least one or many and -----0< to represent zero one or many.
- *Many to many*. We use >------< to represent a many to many association (we can also use the qualifiers shown above >|----0< and …)

Bubble charts
The bubble chart is a commonly used diagram to represent user views. The fields are placed in ovals lines showing the associations (see above).

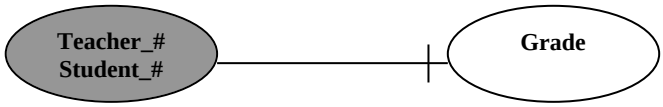
Types and occurrences
An "EMPLOYEE_NAME" refers to a data element (field) and "Fred Jansen" is an occurrence of the "EMPLOYEE_NAME" data element type. A logical data model shows the relations among data types.

Keys and attributes
The data elements in a bubble chart can have tree roles, based on the type of associations drawn on them.

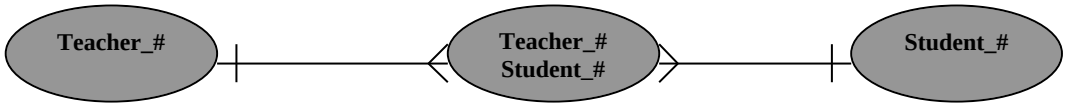
- Attributes. An attribute is a data element whose bubble has NO one to one links leading from it to an other bubble.
- Primary keys. A primary key is a data element whose bubble has one or more one to one links leading from it to an other bubble.
- Secondary keys. A secondary key is a data element whose bubble has NO one to one links leading from it to an other bubble but has one ore more one to many links leading to other bubbles.



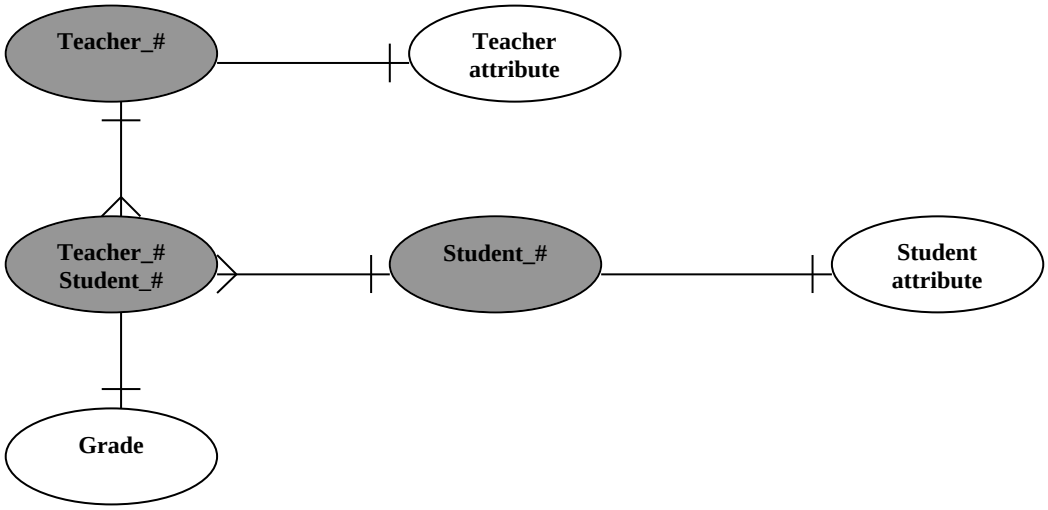
Concatenated keys
It can be to have a data element that cannot be identified by any other single data element. This does not mean that it has no primary key. It may have a primary key made up of more than one data element in combination. Such a key is called a concatenated key.
For example: A teacher may have multiple students, a student may have multiple teachers, and each teacher may give each student a different grade. Therefor, neither the primary key "Teacher_#" nor the primary key "Student_#" identifies the data element (attribute) "Grade". Grade is identified by a concatenated key



Now we place each element of the concatenated key in its own bubble and draw the associations. This process is referred to as exploding the concatenated key.

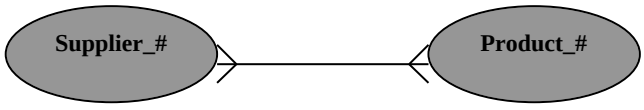


By introducing the concatenated key into the logical data-model, each attribute bubble can always made dependent on a single primary key bubble.



In practice it is sometimes necessary to join together more than two data elements into a concatenated key.

Resolving many to many associations
The requirement for many to many associations may arise in using bubble charts to document logical data models. For example: A supplier may handle many products and each product may be handled by many suppliers.

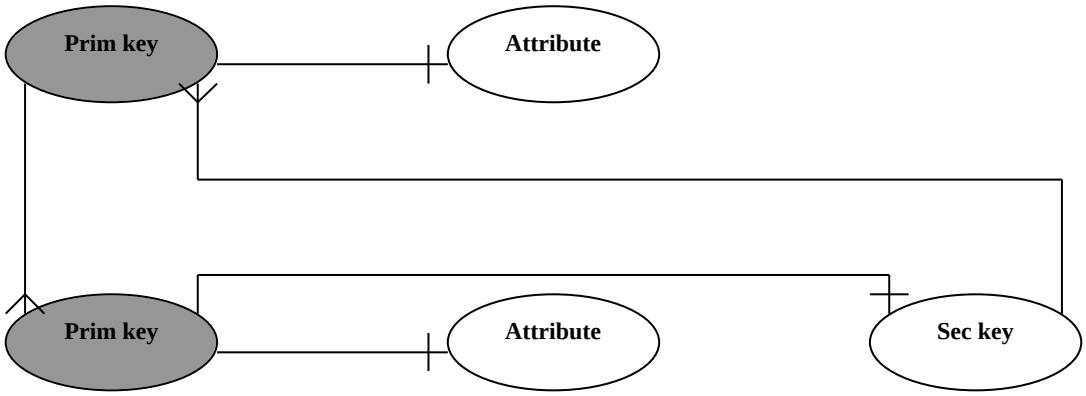


Somme database software is capable of representing this association directly, other software is not. In any case, the inherent properties of data are better represented into a pair of one to one relations and a concatenated key (see Teacher – student example).
Many to many relations rarely – if ever – need to remain in a logical data model, whether or not the target database is capable of representing them directly.

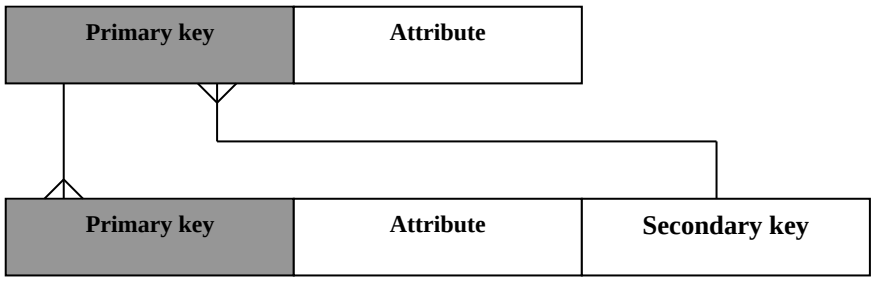
Intersection Data
If we look back to the “Teacher – Student” example above we can see that the data element “grade” cannot be associated with either “Teacher_#” or “Student_#” alone. We need the concatenated key. A data element that requires a many to many association to be resolved by creating a concatenated key is called intersection data.

Remark
Data modeling is an attempt to find the most stable data model. Therefor many to many relations that appear should immediately be resolved by creating and then exploding a concatenated key. In this manner we are not bound to one software and it is easy to add intersection data later without changing the fundamental structure of the logical data model.

Record diagrams
Bubble diagrams show the associations that exist a set of data elements. However if bubble charts grow in size, they often become difficult to read. A record diagram can be used to document the data requirements.



This can be draw as:



To create a record diagram from a bubble chart, we place each attribute and primary key into a box. Then we group all attributes associated with one primary key together with that primary key and we place this key on the left of the group.
Each group that is identified by a primary key is called a logical record. This to distinguish a record in logical data model from whatever may be stored physically on a medium.

Normalization, First Normal Form (1NF)
A table is in first normal form if and only if all fields contain only atomic values; that is, there are no repeating fields in a record.
Example:
Entity book

ID_B	Title	Publisher	Author_1	Author_2	Author_3
12	SQL	O'Reilly	Baxter	Mouse	Peter
13	Zope	CRC	…		
15	Samba	Morgan	Baxter		

In 1NF this would be

ID_B	Title	Publisher	Author
12	SQL	O'Reilly	Baxter
12	SQL	O'Reilly	Mouse
12	SQL	O'Reilly	Peter
13	Zope	CRC	…
15	Samba	Morgan	Baxter

Second Normal Form (2NF)
A table is in the second normal form if and only if it is in 1NF and every nonkey attribute is fully dependent on the primary key.
Consider the "book" entity, "Author" is duplicated many times in the table. If we need to change his name many records must be updated. This is known as the *update anomaly*, and it represents a potential danger.
If want to add a new "Author" but do not now the book-title (not on press yet) we cant. This is know as the *insert anomaly*.
If we delete a book (out of print) we can lose all info about "Author" this anomaly known as the *delete anomaly*.

In 2NF we would get:

ID_B	Title	Publisher
12	SQL	O'Reilly
13	Zope	CRC
15	Samba	Morgan

ID_B	Aut_ID	(Intersection Table)
12	1	
12	2	
12	3	
13	…	
15	1	

Aut_ID	Author
1	Baxter
2	Mouse
3	Peter

Third Normal Form (3NF)
A table is in the second normal form if and only if it is in 2NF and no non-indentifying attributes are depended on other non-indentifying attributes.

In 3NF we would get:

ID_B	Title	ID_P
12	SQL	1
13	Zope	2
15	Samba	3

ID_B	Aut_ID	(Intersection Table)
12	1	
12	2	
12	3	
13	…	
15	1	

Aut_ID	Author
1	Baxter
2	Mouse
3	Peter

ID_P	Publisher
1	O'Reilly
2	CRC
3	Morgan