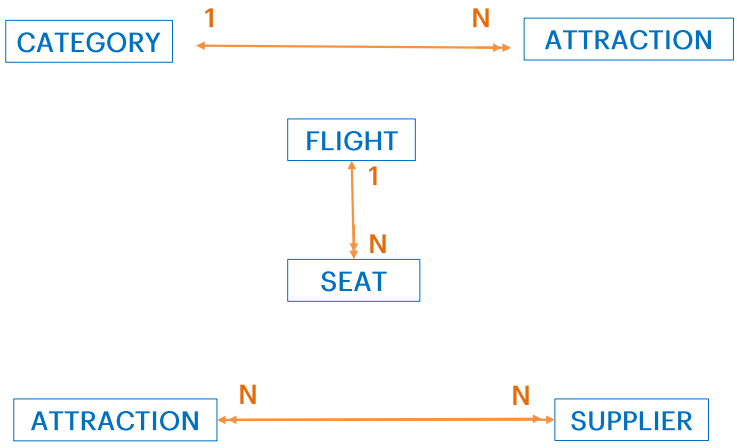
1 to 1 relationships between actors of reality

GeneXus™

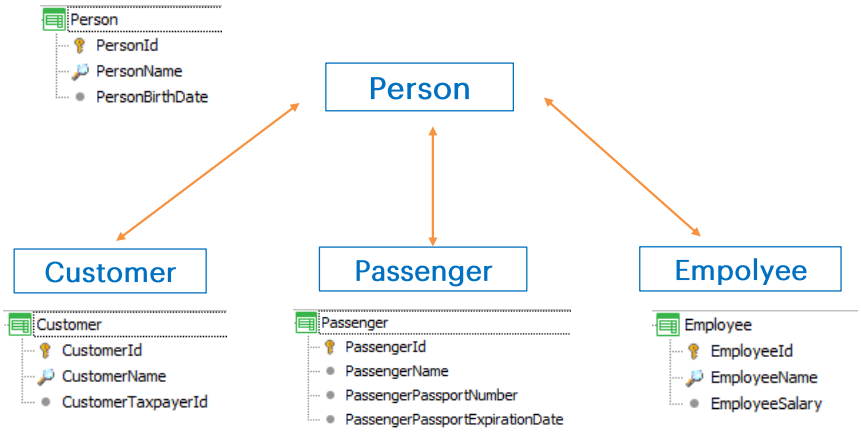


It has been seen that through transactions and their attributes we can represent strong and weak 1 to N relationships between the actors in our reality, as well as N to N relationships.

Now let's focus on 1 to 1 relationships.

In other videos, several cases of representation of 1 to 1 relationships have been mentioned, so let's now review and unify what has already been seen in this video.

Specialization



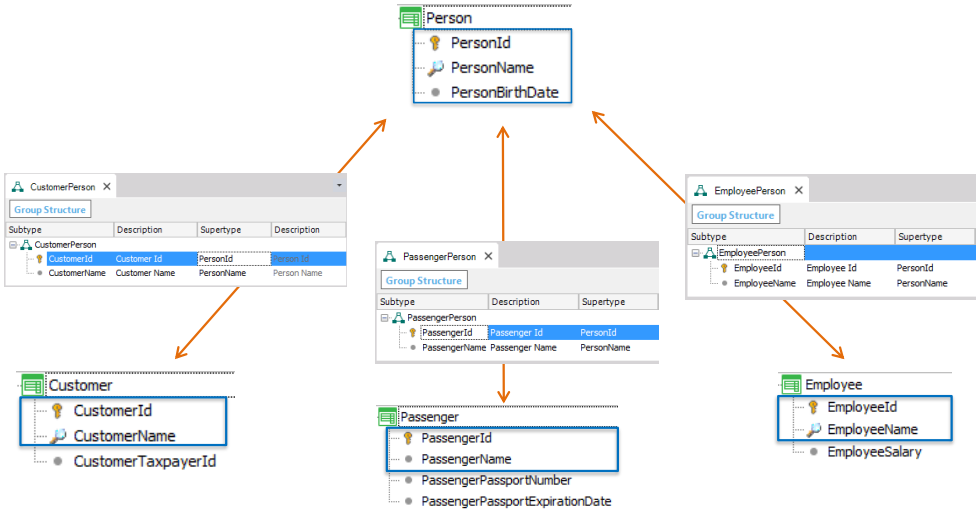
When considering the concept of subtypes, the case of attribute specialization was represented.

In this scenario we have people's general information, such as their Identifier, name and date of birth in the Person transaction, but it is also necessary to record specific information corresponding to Customers, Passengers and Employees, always bearing in mind that they are all Persons.

Every Customer, Passenger, and Employee are Persons.

That is why the Customer, Passenger or Employee's identifier must match exactly that of a Person to represent that they are Persons.

Specialization

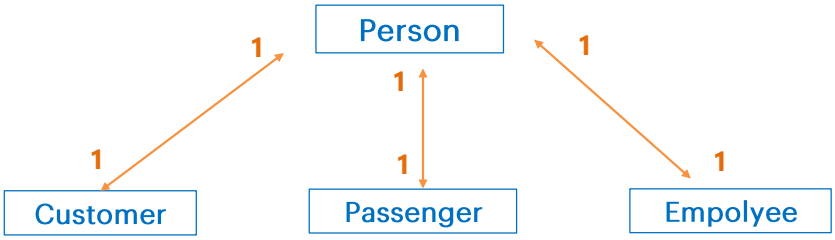


This is achieved by defining the corresponding groups of subtypes.

By doing this, the attributes CustomerId, PassengerId and EmployeeId, besides being the identifiers of the tables Customer, Passenger and Employee respectively, will be, at the same time, foreign keys in the table Person.

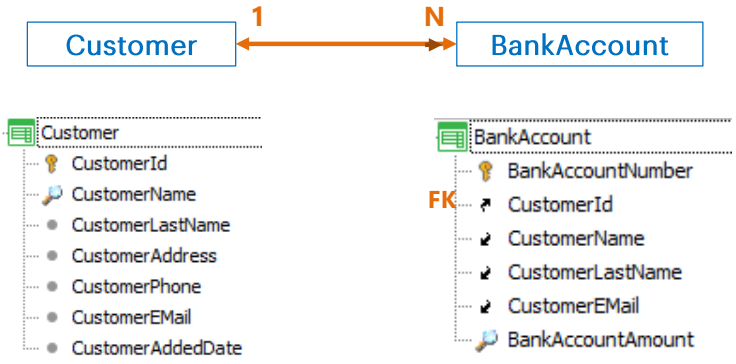
Therefore, when the user enters a value in the identifier of any of these three transactions (Customer, Passenger or Employee), a record with the same value as identifier will be searched for in the Person table.

Specialization



This design then represents 1 to 1 relationships between the general table and the one corresponding to each specialization.

Unique index



Unique index

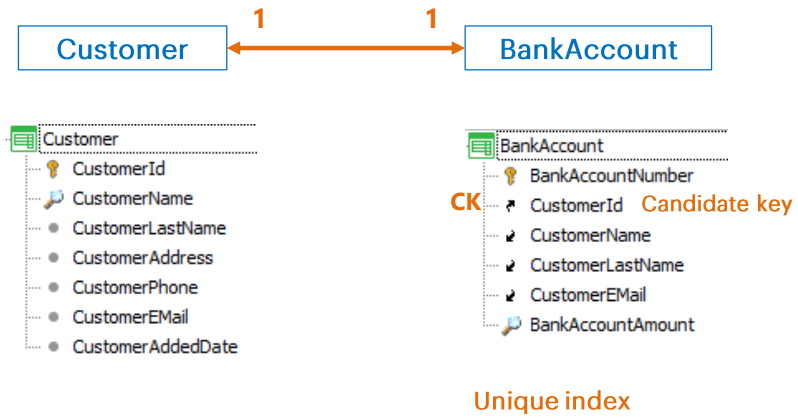
Now let's go over this other scenario. A travel agency needs to associate with each customer the bank account opened to pay for the services hired. In this way, a Customer has only one bank account, and one bank account is associated with only one Customer.

But this design on display represents a 1 to N relationship, and somehow we must then limit "N" to "1."

This attribute, CustomerId, must not be repeated in BankAccount. In other words, we have to make sure that two or more records with the same value cannot exist for that attribute. This means that we want CustomerId to be a candidate key on BankAccount.

How can it be done? By defining a Unique Index.

Unique index



Remember that indexes are efficient ways to access data. A previous video showed that GeneXus automatically creates the primary and foreign indexes in each table, and that you can also create our own indexes and indicate whether they accept duplicate values.

A unique index is an index that does not accept duplicate values. It is a candidate key, an attribute, or set of attributes and, even though it is not the primary key of the transaction, GeneXus will still control its uniqueness automatically.

This design solution then represents that CustomerId is a foreign key in BankAccount, and at the same time its value will not be repeated because it will be automatically controlled by the definition of the unique index.

In this way, a 1 to 1 relationship exists between Customer and BankAccount.

GeneXus™

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications