Transaction Design.

The purpose of this phase is to design the characteristics of known database transactions (applications) in a DBMS-independent way.

A common technique for specifying transactions at a conceptual level is to identify their input/output and functional behavior. By specifying the input and output parameters (arguments), and internal functional flow of control, designers can specify a transaction in a conceptual and system-independent way.

Transactions usually can be grouped into three categories:

(1) retrieval transactions, which are used to retrieve data for display on a screen or for production of a report
(2) update transactions, which are used to enter new data or to modify existing data in the database
(3) mixed transactions, which are used for more complex applications that do some retrieval and some update.

For example,

(1) Check Class Status (out Class.Status)

(2) Create Course (in Course.Id, in Course.Content, in Course.Department, in Course.Prerequisites, in Course.Standards)

(3) Enrolls in Class (in Schedule, in Class, in Course, out Class, out Bill, out Schedule)

Or you can include the transaction in the middle, like

Enrolls in Class (in Schedule, Check Class, Update Class, Update Schedule, Generate Bill, out Schedule)

Note that in the example (3) the functional flow of control requires that first we have the Schedule (to see that there is no conflict with the time/date of the class) before we check the Class, and then we update the Class, and finally we update the Schedule, after updating the Bill.
In a pictorial way, we can represent the example 3 as follows: