# Building a Better Data System: What Are Process and Data Models?

March 2016

Robin Nelson
Bruce Bull

**The DaSy Center**

DaSy The Center for IDEA Early Childhood Data Systems

The contents of this report were developed under a grant from the U.S. Department of Education, #H373Z120002. However, those contents do not necessarily represent the policy of the U.S. Department of Education, and you should not assume endorsement by the Federal Government. Project Officers, Meredith Miceli and Richelle Davis.


March 2016

**Suggested citation:**
Nelson, R., & Bull, B. (2016). *Building a better data system: What are process and data models?* Menlo Park, CA: SRI International.

# Contents

# Introduction

This document provides IDEA Part C and Part B 619 staff involved in data system development with an overview of process modeling and data modeling and explains the value of each in the development or major enhancement of data systems. Process modeling and data modeling are important activities in data system development and are particularly valuable for creating business requirements. (For more extensive information on both types of modeling, readers are encouraged to access the references and resources provided in hyperlinks throughout this document.)

Process and data models are effective communication tools that foster understanding among content experts, users, and information technology (IT) staff.

- A process model illustrates the processes the data system is to support.
- A data model illustrates the information the data system is to manage.

Process and data models contribute to successful data system development by informing requirements analysis. Both are important aspects of the System Design and Development subcomponent of the Data System Framework, particularly Quality Indicators 2 and 3. Requirements analysis involves analyzing, detailing, prioritizing, and validating the business[1] requirements, that is, the program functions that the data system must provide for and support. Poorly defined requirements often have real and significant impacts on the development of a data system, resulting in delays, instability, poor quality, and the need for costly redesign work. Two problems that are common in requirements analysis are lack of understanding of the program content by IT staff and miscommunication between program staff and IT staff. Process and data models are visual representations, and are tools that can be used to clearly analyze and communicate the needs of the program in a non-technical manner. Throughout this document we use both non-educational and Part C/Part B 619 examples of process and data models.

# What Is a Process Model?

A process model is a formal way of visually representing how a business system operates, including the activities that are performed and how information moves among them. The model depicts the functions and operations (i.e., processes) of the program that will eventually be performed by the data system being developed (NY State Guidebook: System Requirements Analysis). At a basic level, process models identify both an inventory of processes and the relationships and sequencing of those processes within the data system as it supports the program. The goal of process modeling is to identify and visually depict processes the data system supports.

Process modeling typically starts with identification of the major processes of the business or program, such as child find and referral processes, eligibility determination, and IFSP/IEP (Individualized Family Service Plan/Individualized Education Program) development for the Part C or Part B 619 program. These major processes are diagrammed in the process model and are broken down into manageable subprocesses until no further breakdown is feasible. For example, the major process of IFSP/IEP development could be broken down into developing an initial IFSP/IEP, reviewing an existing IFSP/IEP, and revising an IFSP/IEP. Each of these processes would be further broken down. Processes such as developing IFSP/IEP outcomes, identifying planned services and supports, and determining the planned frequency, duration, and setting of those services would all be shown as subprocesses in the completed process model.

There are different kinds of process models and different techniques for doing process modeling (Business Process Modeling Techniques, Process Modeling Techniques). Flow charts, business modeling,

[1] Generally, in data systems work, the term "business" does not refer solely to the accounting or financial aspects of a program, but rather more broadly to any function of the program (Part C or Part B preschool) that requires the use of data.

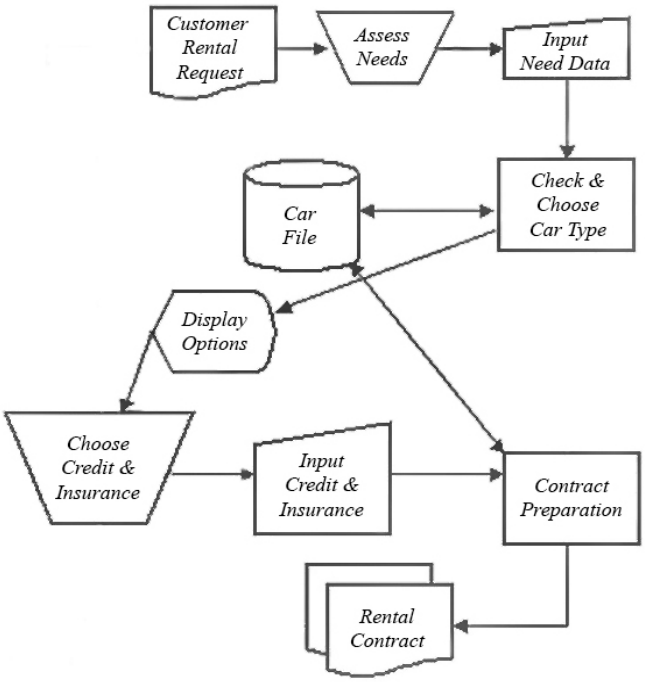
data flow diagrams, and work flow diagrams are all types of process modeling. Regardless of the type or technique, the important elements of any process model include defining and addressing

- the scope of the process, that is, where a process begins and ends;
- the desired outcome of the process;
- the steps or activities involved in the process;
- the flow or sequence of steps through both the most common paths of the process and the variations or exceptions in the process; and
- the inputs (what is needed to start the process) and the outputs (what is generated as a result of the process).

It is very important to define and address variations or exceptions to a process because of their impact on requirements analysis. For example, in Part C the process of eligibility determination might appear very different for children with an established condition based on a diagnosis from a pediatrician than for children with a suspected developmental delay. The process model must depict (and the data system must be able to support) the common processes as well as any and all less common variations.

Exhibit 1 provides an example of a simple process model of a car rental organization. Starting from the top left, the process is shown as a sequential set of steps that must occur before a customer actually obtains a rental contract in order to rent a car. The customer initiates the process by requesting the rental (the input), and the process ends when the customer has the rental contract in hand (the output). In this example and in all process models, each type of symbol has meaning. Different symbols represent data received, a decision point, a document secured, and the like. (For more information on symbols and how to draw process models, go to Flowchart Guide.)

Exhibit 1. Car Rental Process Model



When programs attempt to streamline processes to achieve greater efficiency, an *as is* process model is often developed to represent the current process. Then a *to be* model is developed to represent the new, revised process. Creating both *as is* and *to be* process models is an excellent opportunity to communicate across program and IT staff the limitations of the existing system and the expectations of a new or enhanced system.

One common mistake in developing process models is that content experts are prone to too much discussion on the "how"—how procedures should be accomplished. This may come at the expense of efficiently developing the process view of the model (the "what"). A procedure (how) captures specific actions (such as clicks, downloading files, updating files) that business users will do—it is how they do their job. These are actions one should find in an operating manual or training guide. In contrast, a process (what) captures the sequence of steps at a slightly higher level. The process focuses on what needs to be accomplished, the sequence or flow of information throughout the steps, and how activities from different aspects of the program are related. Maintaining communication and effort to explain and depict higher level processes—and not getting bogged down in procedures—will expedite the creation of the process model.

In the context of business requirements analysis, using process models can facilitate good communication between program staff and IT staff. A process model helps those who are technology oriented acquire knowledge about the program instead of focusing exclusively on the technical aspects of the data system. A process model is a tool that can be used by both groups to analyze and communicate what the program wants without getting too deep into technical details. Furthermore, a process model, developed in the natural language of program staff, helps IT staff learn about the program needs as processes are described. A process model also serves as a common and shared representation of the program. Among program staff, process modeling can foster understanding should different staff members have different conceptions of program processes.

## What Is a Data Model?

A data model is a visual representation of the information requirements of an organization or program. It is a schematic showing the organization and structure of the data. The goal of a data model is to make sure that all the entities required by the database are included and completely and accurately represented. The data model

- identifies all unique database entities used or produced by the data system;
- captures all the characteristics of those entities (data attributes); and
- describes the relationships between the entities as they align within the program.

An *entity* is something that exists either physically or logically—it can be thought of as a noun. An entity may be a physical object such as a car, employee, or IFSP/IEP; an event, such as a car service or transition conference; or a concept, such as a car rental transaction or service delivery model. An *attribute* is a property or characteristic that describes an entity. In the car rental process model, entities are the customer, car, and rental contract. Attributes of a car would include make/model, year, and size (full size, midsize, compact). In an example of a child as an entity, attributes of a child would include first name, last name, date of birth, unique child identification number,[2] etc.

Once the data entities and their respective attributes have been defined, relationships between them are addressed in the data model. A *relationship* reflects the association or link between entities. There are several dimensions used to describe relationships, including the number of entities in the relationship and the direction of the relationship (Relationship Types). One important dimension of relationships is based on the number of entities to which another entity can be associated in a relationship set. Exhibit 2 presents

[2] For examples of common entities and attributes used in early childhood, see the Common Education Data Standards.

the different types of these relationships, definitions of each, and two examples for each type of relationship.

Exhibit 2. Types of Relationships Among Database Tables

| Relationship | Description | Car Example | Part C/Part B 619 Example |
|---|---|---|---|
| One to one | A row (record) in a table is associated with one and only one row in another table. | One car is associated with only one vehicle identification number. | One child is associated with only one child identification number. |
| One to many | A row in a table in a database can be associated with one or many rows in another table. | One car make (e.g., Toyota) is associated with many vehicle models, (Camry, Corolla, Prius, Tundra). | One IFSP or IEP is associated with many types of services. |
| Many to many | Rows in one table can map to multiple rows in the second table, and those rows in the second table can also map to multiple (different) rows in the first table. | Car dealers sell many/different car models, and many car models are sold by many/different dealers. | Service providers serve many/different children, and children receive services from many/different service providers. |

These types of relationships are very important in modeling the database structure. Entities are tables in the database, and entity data—i.e., attributes—are stored as columns in the data tables. The relationship of one data table to another (one entity to another) must be precise in order to explain how each table is related to other tables within the database, allowing for appropriate data linkages and subsequent analyses.

Although a full exploration is beyond the scope of this document, data models can be represented in one of three ways, with increasing levels of detail.

- **Conceptual data models** are the simplest and most high level, and they feature entities and entity relationships.
- **Logical data models** include entities, relationships, and attributes, plus unique identifiers that help with linking data tables by the use of keys.[3]
- **Physical data models** depict how data are represented in the database and include table names, column names, column data types, and keys.

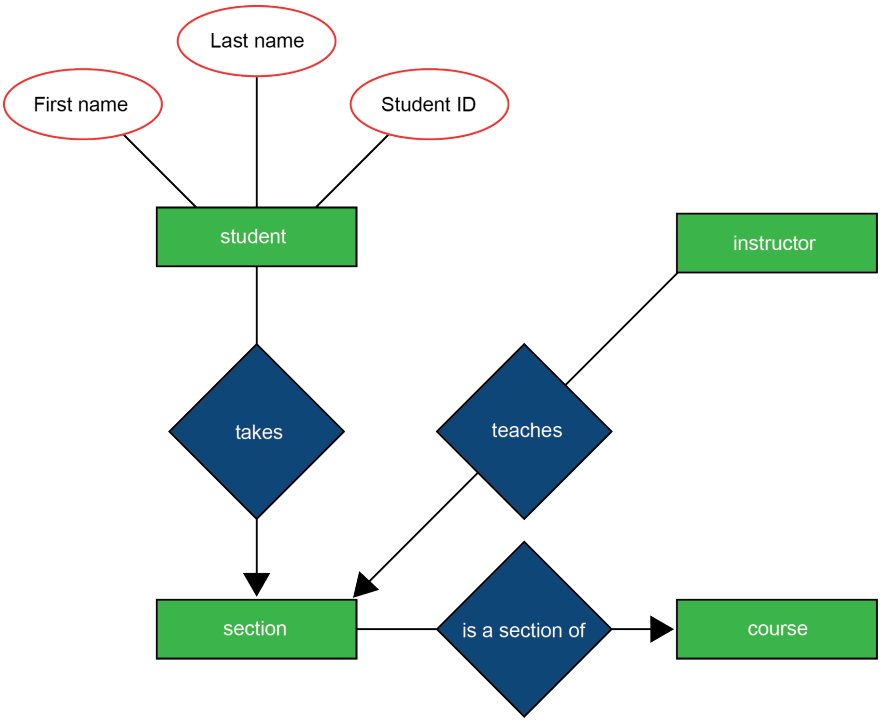
## What Is an Entity-Relationship Diagram (ERD)?

An entity-relationship diagram is a graphical representation of entities and their relationships to each other, and it can be used to display conceptual, logical, and/or physical data models. The power of the ERD is its ability to display those relationships clearly and accurately. Developing anything but the simplest database without an ERD is like building a house without a building plan.

Exhibit 3 shows a simple ERD with four entities in the green rectangles—student, instructor, section, and course—and the relationship between those entities in the blue diamonds. The attributes of a student are diagrammed using circles extending from the student entity to each student attribute: first name, last name, and student ID.

[3] Some columns in database tables are special columns because they identify a particular record in a table. The special columns are called keys and are used to join tables. The two most common keys are primary keys and foreign keys. The primary key uniquely identifies each record in the table and may consist of a single attribute or multiple attributes in combination. A foreign key is a field that matches the primary key of another table and can be used to cross-reference tables.

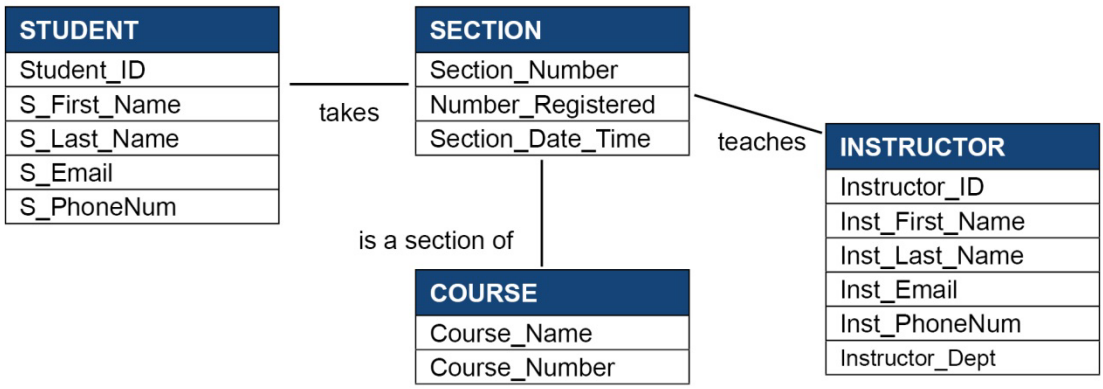
Exhibit 3. Simple Entity-Relationship Diagram (ERD)




The logical data model with its defined data entities, their attributes, and relationships becomes the foundation of the physical data model (the actual database). The data model is the "blueprint" and displays the entities as tables in the database and attributes as columns within the tables. IT staff use the data model to build the physical database, define the relational tables, and define special columns, or keys, that will be used to join tables in the physical database.

Exhibit 4[4] shows a slightly more complex ERD than the one in Exhibit 3. It shows aspects of the physical data model, e.g., entities as tables of the database. The table names are in blue followed by a list of attributes.

Exhibit 4. Entity-Relationship Diagram: Entities as Database Tables




[4] Although ERDs typically identify keys as well as the types of relationships between the entities, for the sake of simplicity that content is not depicted. For more information on ways to notate keys and types of relationships, see *ERD Symbols*.

In summary, the ERD serves as a good reference and documentation tool that can

- help to define the entities and attributes in the easily understood natural language of the program;
- allow nontechnical users and program staff to more easily review and verify the model;
- provide an effective communication tool for discussions across IT and program staff; and
- support the development of a data dictionary.

If the data model is wrong, the data system will not do everything that users want and expect it to do: Needed data for critical reports may be omitted, data may not be able to be linked for certain analyses, or results may be produced that are incorrect or inconsistent.

## Example of a Process Model and Data Model: Child Outcomes Data

Child outcomes data are required for the IDEA Annual Performance Report (APR)—indicator 3 for the Part C APR and indicator 7 for the Part B APR. States are required to report on the percentage of infants and toddlers with IFSPs or preschool children with IEPs who, upon leaving services, demonstrate improvement in the following:

1. Positive social-emotional skills (including social relationships)
2. Acquisition and use of knowledge and skills (including early language/communication [and early literacy])
3. Use of appropriate behaviors to meet their needs.

The vast majority of states use the Child Outcomes Summary (COS)[5] process to produce these data. States using the COS process collect child-level data at entry and again at exit to reflect a child's functioning relative to same-age peers for each of the three outcomes listed above.

### Example of Child Outcomes Process Model

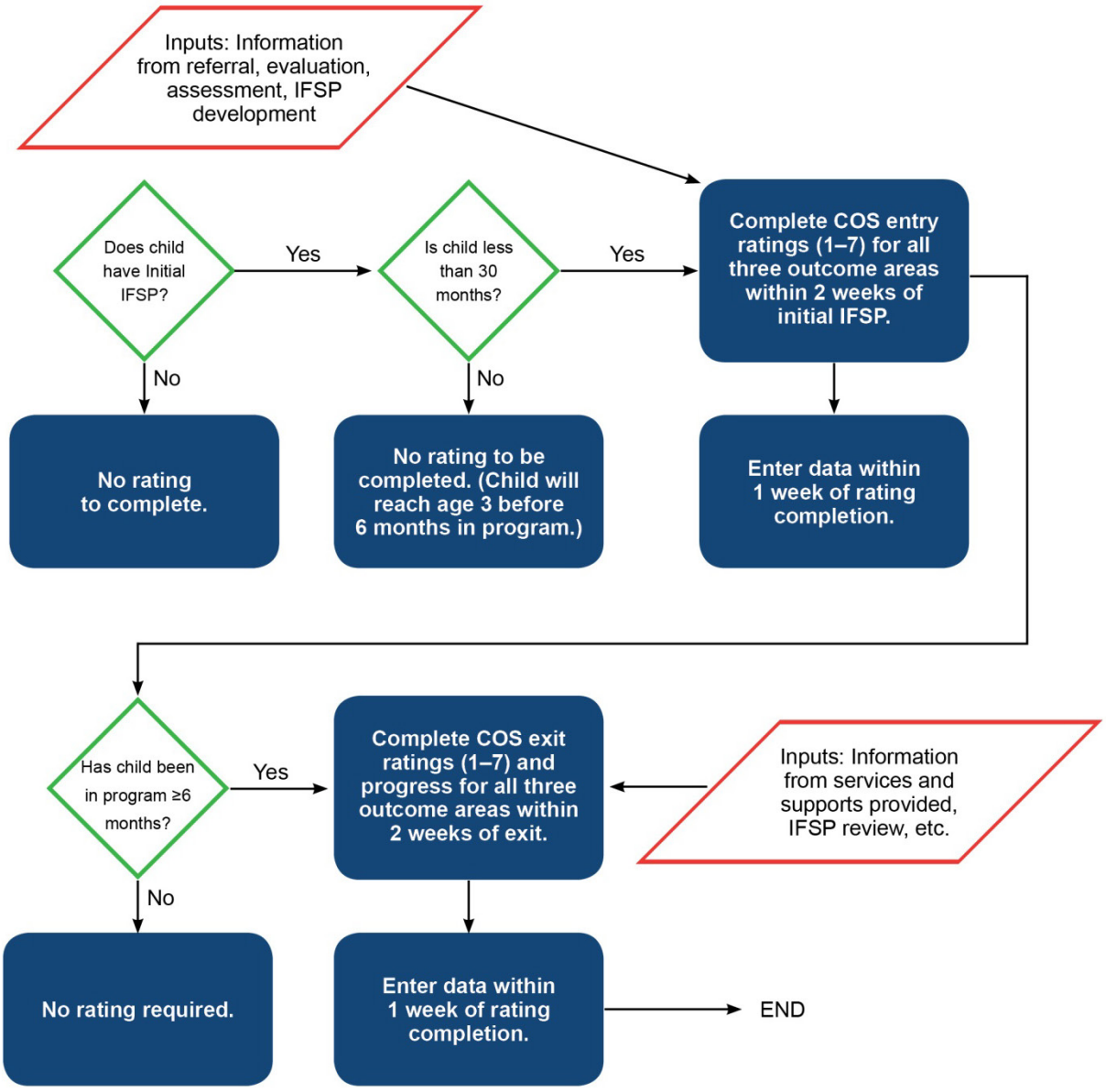
The child outcomes process model (Exhibit 5) reflects processes that occur after referral and after the determination that the child is eligible for the Part C (or Part B 619) program. The model starts at the time the child becomes enrolled in services, that is, when the child has an initial IFSP (or IEP). Although states that use the COS process may vary slightly in their approach to collecting COS data, most would use information gathered during those earlier processes (i.e., referral, evaluation, assessment, and IFSP development) to inform the team decision for the COS entry rating. The information (or data) acquired in those earlier processes is represented by a data file symbol (shown in Exhibit 5 as a box with a red border). Similarly, information gathered during ongoing assessment, services and supports, and IFSP reviews would be used to make the exit ratings. The COS process model for entry and exit ratings should focus on the processes that must occur (not how they occur) and thus might look like Exhibit 5.

Note that the process model includes some time requirements (e.g., within 2 weeks of exit, within 1 week of rating completion). This information may help IT staff incorporate edit checks, alerts, and other local end-user functionality into any state-supported data system developed for local program use.

Exhibit 5 illustrates typical processes for use of the COS, but individual states may have different policies and may require state-specific models. For example, some states require entry ratings for all children regardless of their age at entry, and some states collect data at additional points in time, such as every 6 months or annually, which are not required for federal reporting. Ratings at these additional points would need to be reflected in the process model (and data model), as applicable. In addition, any such exceptions to the standard rating process should be noted. For example, if the process differs for children who exit one local program and transfer to another (i.e., who are not new referrals), that should be depicted in the model.

[5] For information about child outcomes and the COS process, see http://ectacenter.org/eco at the Early Childhood Technical Assistance Center website.

Exhibit 5. Example of a Process Model for Using a Part C COS Process




*Note*. Example shows the data collection process for a **new** referral from entry to exit from Part C services.

## Example of Child Outcomes Data Model

Exhibit 6 is a sample database table for child outcomes data. There are fields for the rating for each outcome, and the *Time_Code* field identifies whether the rating is at entry or exit. There are also calculated fields in the table for the progress categories (a through e). These are automatically generated by the database and are used to compute the summary statements.

Exhibit 6. Example of a Database Table for Child Outcomes Data

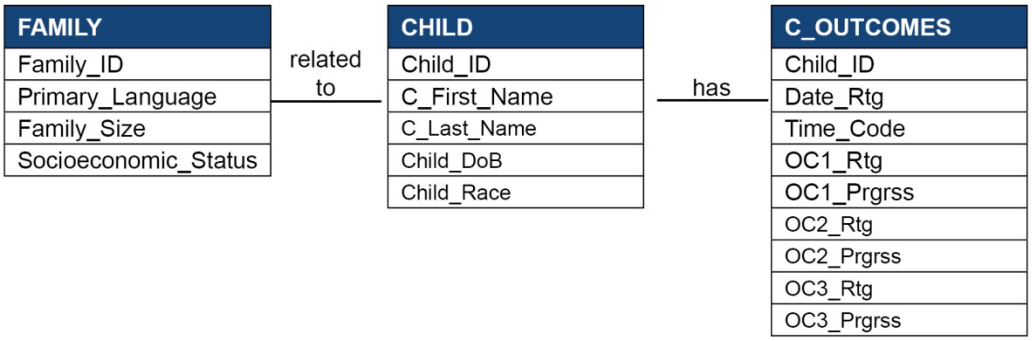
| | | |
|---|---|---|
| **Key** | **Child** | Child_ID – ID associated with a particular child |
| | **Prog** | Prog_ID – ID associated with a particular program |
| | Date_Rtg | Date of the outcome ratings |
| | Time_Code | Code for this rating time period (1, 2) |
| | Time_Dcd | This rating time period decoded:<br>1 = Entry<br>2 = Exit |
| | OC1_Rtg | Outcome 1 Rating (1–7) |
| | OC1_Prgrss | Outcome 1 Progress at Exit (Y or N) |
| | OC2_Rtg | Outcome 2 Rating (1–7) |
| | OC2_Prgrss | Outcome 2 Progress at Exit (Y or N) |
| | OC3_Rtg | Outcome 3 Rating (1–7) |
| | OC3_Prgrss | Outcome 3 Progress at Exit (Y or N) |
| | OC1_Cat | Calculated progress category for outcome 1: (a-e) |
| | OC2_Cat | Calculated progress category for outcome 2: (a-e) |
| | OC3_Cat | Calculated progress category for outcome 3: (a-e) |

Various business requirements, in the form of edit checks or logical consistency checks, could also be put in place for these data. Here are some examples: the entry outcome rating may only be a number between 1 and 7, nothing else; the system, which calculates the progress category upon processing the exit rating and the yes/no progress question, does not allow values for calculated fields that result in illogical data; the dates for the ratings are restricted to not allow a date before the child's enrollment in the program, a date after the child turns 3, or a future date.

If child outcome data are housed within a comprehensive data system, states might also have data about characteristics of the child (e.g., gender, race/ethnicity, date of birth), information about the family (e.g., primary language, family size, household income or socioeconomic status); eligibility information (e.g., reason eligible, type extent of developmental delay); and other information (e.g., date of referral, date of enrollment). These data are not part of the COS process; however, in a comprehensive data system child-level COS data would be linked to the child and family characteristics through keys, as discussed above.

Exhibit 7 shows a very simple ERD with a table for child characteristics, family characteristics and child outcomes.

Exhibit 7. Example of ERD for Child and Family Characteristics and Child Outcomes Data




## Summary

Process modeling and data modeling are two activities critical to successful data system development or enhancement. The process model visually represents all the business processes that interact with the system. The data model depicts all the data requirements of the system: the entities, their attributes, and the relationships between the entities as aligned within the program area. There are many different methodologies and techniques available to do both kinds of models, and different techniques have advantages and disadvantages. In general, however, the use of process and data models can facilitate good communication between program staff and IT staff and can be used to analyze and communicate what the program wants from the new or enhanced data system.

Both process and data models, if developed correctly, will be useful in defining the full set of business requirements for the data system. In fact, it may be necessary, and is often helpful, to simultaneously generate process and data models when defining data system business requirements. By developing process and data models together through the analysis of the business requirements, the models can be updated to reflect updated business requirements and the business requirements can be refined to reflect process and data modeling. Both program staff and IT staff can and should work together to reconcile business requirements with the process and data models. Program staff and IT staff working together on process models, data models, and business requirements substantially increases the likelihood that the new or enhanced data system will function in accordance with their vision for it.