Resources

The Resources that are used by an Ed-Fi REST API are compositions of entities, attributes, and associations. Resources are domain aggregates that have been identified from the Ed-Fi Unifying Data Model (UDM) according to the principles of Domain-Driven Design (DDD). Use cases and events in the domain typically center on individual domain aggregates.

Domain aggregates are organized along transactional boundaries, where the data contained should “live” and “die” together. For example, the Discipline Incident domain aggregate contains details about a discipline incident, and also captures data related to the behaviors and students involved. A Discipline Action is not generally captured at the same time as the Discipline Incident. Therefore, each is separated into its own domain aggregate.

Each domain aggregate has an aggregate root. An aggregate root is an entity (and in some cases an association) that includes other entities, their attributes, and associations. The subordinate entities, attributes, and associations of a domain aggregate are not directly accessible and can only be referenced through the aggregate root.

Most entities in the Ed-Fi UDM are aggregate roots (e.g., Student, School, Course). They contain no other entities. In some cases, an association that represents a significant domain concept is also represented as an aggregate root. For example, the StudentSchoolAssociation is represented as an aggregate root because it reflects enrollment.

In the table below, the domain aggregate for a Course is constructed from a number of course-related entities in the Ed-Fi UDM. A complete list of Resources, Types, and Descriptors is included in the API Resource Listing.

Table 1. Sample Domain Aggregate

Domain Aggregate	Entities
Course	Course
	CourseCompetencyLevel
	CourseGradeLevel
	CourseIdentificationCode
	CourseLearningObjective
	CourseLearningStandard
	CourseLevelCharacteristic

Each resource exposed by an Ed-Fi REST API *must* be referenced by an internally-assigned universally unique identifier (UUID). While the specific algorithm for generating these identifiers is not prescribed in these guidelines, the identifiers *should* be generated using a UUID implementation such as Microsoft’s GUID (globally unique identifier)[5]. An Ed-Fi REST API *should* generate unique identifiers for its clients, and *should not* accept client-generated identifiers when inserting or updating data.

All Resources *must* be created with and also be retrievable by one or more externally defined primary key values. Those values *must* be natural keys of the resource. For example, a Session is uniquely identified by the Session Name, Term, and School Year. Resources *must* be accessible by primary key values using the standard HTTP GET query string search syntax:

```
{resourceURI}?keyField1={value1}&keyField2={value2}
```

PUT, PATCH, and DELETE operations *must* be identified using their UUID (i.e., `{resourceURI}/{id}`). PUT, PATCH, and DELETE operations *should* also be identifiable using their primary key values (natural keys).

Resource Extensions

The Ed-Fi UDM can be extended to modify the structure of existing entities or create entirely new entities.[6] In the context of an Ed-Fi REST API, these are called resource extensions. Consumers of an Ed-Fi REST API interact with resource extensions just as they interact with other Resources. For example, if the Student resource has been extended in the data model supported by the API platform host, an API consumer requesting a student will receive the extended resource.

The same practices used for native Ed-Fi REST API interchanges and Resources should be used for extensions. Extension Resources and extension entities on Ed-Fi standard interchange schemas should follow the same conventions as outlined in the Ed-Fi Core XML Schema. Extension attributes (that is, new attributes for existing Resources) do not carry a suffix identifying them as an extension, and are indistinguishable from native attributes when viewed in the API.

REST API Specification Documentation Contents

Find out more about the Ed-Fi REST API:

Types and Descriptors

Types and Descriptors exposed through the API are analogous to enumerations in XML or lookup values in a database. The primary difference is that Types are generally immutable in the context of particular Ed-Fi technology implementations because they are intended to form a relatively stable part of the data model, while Descriptors are elements designed to support change.

Accordingly, there are differences in how these similar structures are supported through the API.

Types *must* be read-only from the API's perspective (as they are analogous to enumerations).

Descriptors are typically read-only from the API's perspective; however, an implementation *may* allow them to be created using POST and updated using PUT.

[5] For additional information on UUIDs, see here.

[6] Extensions to the Ed-Fi data model in an API context are discussed in the Ed-Fi ODS / API documentation*,* available here.