

information system design and software engineering lab

ENTITY RELATIONSHIP DIAGRAM



TEACHER STUDENT COLLABORATION SYSTEM

CSE3224

January 25, 2018

sUBMITTED bY

MAHIR RASHID || 15-01-04-011

TASNIM MASHRUR || 15-01-04-013

ATIQUL ISLAM || 15-01-04-014

MD. SAIFUL ISLAM || 15-01-04-027

**ENTITY-RELATIONSHIP DIAGRAM (ERD):**

The entity-relationship model (or ER model) is a way of graphically representing the logical relationships of entities (or objects) in order to create a database that shows the relationship between people, objects, places, concepts or events within that system. An ERD is a [data modeling](http://searchdatamanagement.techtarget.com/definition/data-modeling) technique that can help define business processes and can be used as the foundation for a [relational database](http://searchsqlserver.techtarget.com/definition/relational-database).

**The elements of an ERD are:**

* Entities
* Relationships
* Attributes

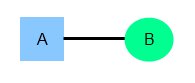
**Uses:**

* Entities are the "things" for which we want to store information. It is a piece of data-an object or concept about which data is stored. An entity is a person, place, thing or event.
* A relationship is how the data is shared between entities. There are three types of relationships between entities:

**1.One-to-One:**

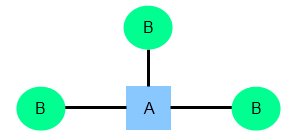
A one-to-one relationship is a type of cardinality that refers to the relationship between two entities (see also entity–relationship model) A and B in which one element of A may only be linked to one element of B.

For example, in a database of employees, each employee name (A) is associated with only one social security number (B).



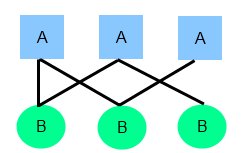
**2. One-to-Many:**

One instance of an entity (A) is associated with zero, one or many instances of another entity (B), but for one instance of entity B there is only one instance of entity A. For example, for a company with all employees working in one building, the building name (A) is associated with many different employees (B), but those employees all share the same singular association with entity A.



**3. Many-to-Many:**

Many-to-Many relationship is a type of cardinality that refers to the relationship between two entities A and B in which A may contain a parent instance for which there are many children in B and vice versa. For example, for a company in which all of its employees work on multiple projects, each instance of an employee (A) is associated with many instances of a project (B), and at the same time, each instance of a project (B) has multiple employees (A) associated with it.



**Attributes** are the data we want to collect for an entity.

There are different types of attribute:

1. **Optional attributes:**

A required attribute is an attribute that must have a value in it, while an optional attribute may not have a value in it and can be left blank. The reasoning for making an attribute required is to ensure that data are collected for that specific characteristic.

2. **Keys and non-keys Attributes:**

In every entity an attribute or grouped attributes uniquely identify that entity. These attributes are the key attributes and range from Primary key (single attribute identifier) to a Composite Key (Multi attribute Identifier). The rest of the attributes after the identifier are considered the non-key attributes or descriptors, which just describe the entity.

1. **Simple and Composite Attributes:**

Attributes can be classified as having many parts to them or a single unbreakable attribute. The composite attribute is an attribute that can be subdivided into other single attributes with meanings of their own. A simple attribute is just an attribute that cannot be subdivided into parts.

1. **Single-valued and multi-valued Attributes:**

Attributes can be classified as single or multivalued. The single-value attribute can only have one value, while the multi-valued attributes usually can store multiple data in them.

1. **Derived Attributes:**

The last category that attributes can be defined is called a derived attribute, where one attribute is calculated from another attribute. The derived attribute may not be stored in the database but rather calculated using algorithm.

**ENTITY SET NAMES:**

1. Department

2. Courses

3. Teacher

4. Materials

5. Students

6. Quizes

7. Attendence

|  |  |
| --- | --- |
| **DEPARTMENT** | |
| Dept name  Teacher\_id  Student\_id | Varchar (5000)  Varchar (5000)  Varchar (5000) |

**ATTRIBUTES OF EACH ENTITY:**

|  |  |
| --- | --- |
| **COURSES** | |
| Course no  Course\_name  Teacher\_id  Student\_id | Varchar (5000)  Varchar (5000)  Varchar (5000)  Varchar (5000) |

|  |  |
| --- | --- |
| **TEACHER** | |
| Teacher id  Teacher\_name  Teacher\_mail  Teacher\_cell\_ no | Varchar (5000)  Varchar (5000)  Varchar (5000)  Varchar (5000) |

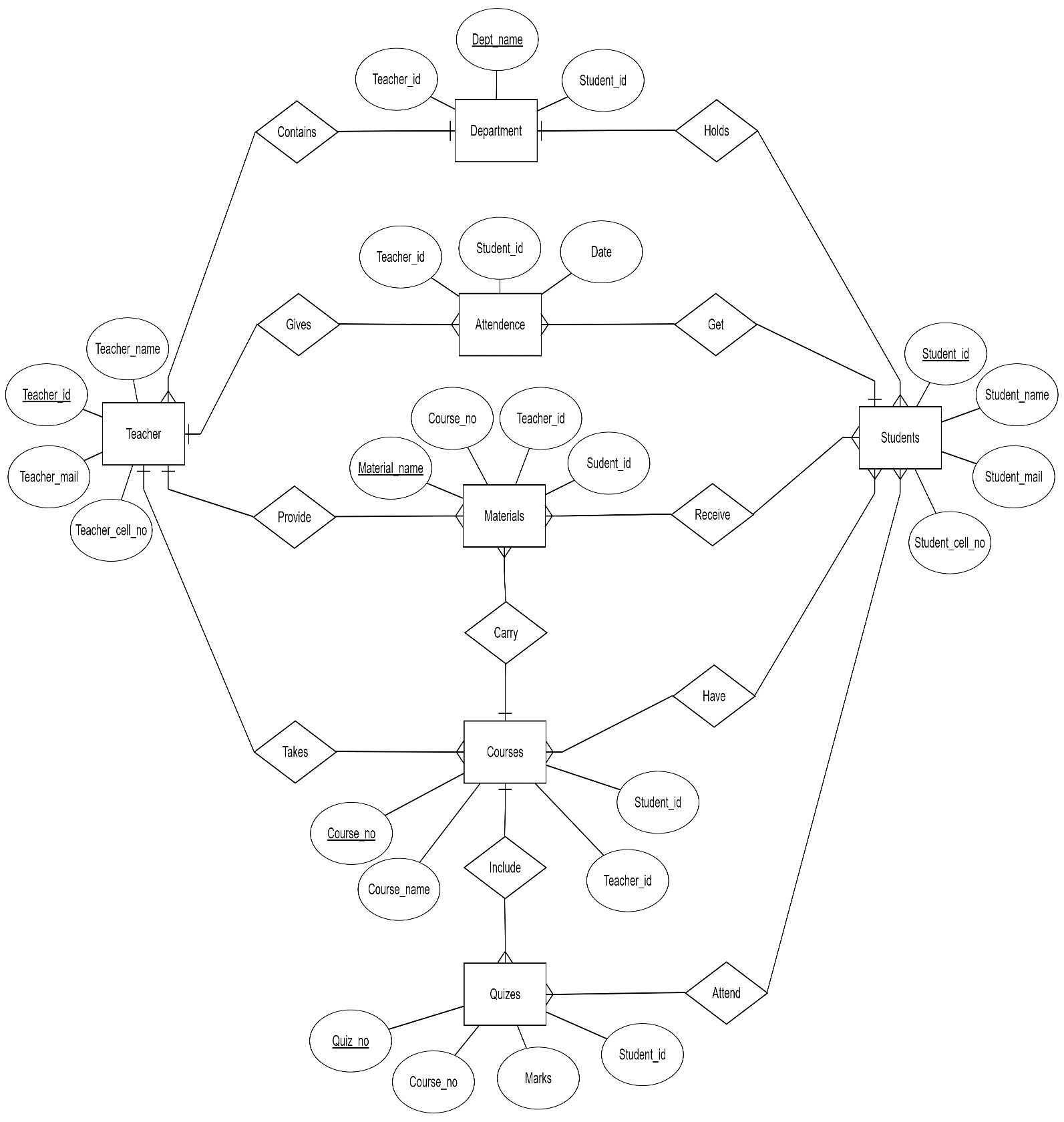
|  |  |
| --- | --- |
| **Materials** | |
| Material\_name  Course\_no  Student\_id  Teacher\_id | Varchar (5000)  Varchar (5000)  Varchar (5000)  Varchar (5000) |

|  |  |
| --- | --- |
| **STUDENTS** | |
| Student\_id  Student\_name  Student\_mail  Student\_cell\_no | Varchar (5000)  Varchar (5000)  Varchar (5000)  Varchar (5000) |

|  |  |
| --- | --- |
| **QUIZES** | |
| Quiz\_no  Course\_no  Marks  Student\_id | Varchar (5000)  Varchar (5000)  Varchar (5000)  Varchar (5000) |

|  |  |
| --- | --- |
| **ATTENDENCE** | |
| Teacher\_id  Student\_id  Date | Varchar (5000)  Varchar (5000)  Varchar (5000) |

**Entity Relationship Diagram:**

**Conclusion:** We will try our best to implement the project as the ERD and Class Diagram we have prepared. Hopefully we can develop a useful, user friendly collaboration system that satisfies the requirements of our client.