# **Chapter 1: Project Introduction**

Our Project is to critique the current choice of the University to keep a spreadsheet-based data collection and storage system for tracking research productivity of its faculty, and to design a 3NF relational database to hold this information in. We will also be creating an SQL scribe that generates our database schema and design a python program to extract data from our database.

There are many reasons that the current spreadsheet-based data collection and storage system is vastly inefficient and inferior to a database-based solution. One reason is that for a university tracking research productivity of all it’s faculty members, it will scale horribly compared to a database-based solution. The more faculty that the university employs and requires research from, the more the spreadsheet-based data collection and storage will become inefficient, since a different excel spreadsheet is needed for each employee whereas in a database you will only need one table. Another reason that shows the inefficiency of the current spreadsheet-based data collection system is that spreadsheets do not have flexible query languages where we can submit queries in order to retrieve information from them quickly and easily, they will have to be manually retrieved. In addition, formatting and appearance of the spreadsheet may sometimes be used to apply meaning to the data such as color coding, which can create confusion as any data should be independent from formatting.

In contrast, a database-based solution would be superior to a spreadsheet-based database. One example is that in contrast to spreadsheet-based data collection, a database is extremely scalable in comparison. No matter how many employees the university hires in the future, the database will be able to comfortably accommodate them and there will be no need to create a brand new spreadsheet for the employee. Another advantage is that a database-based solution has access to query languages which allows us to retrieve specific data from them quickly without having to open multiple spreadsheets and manually find the information ourselves.

# **Chapter 2: Database Design and Implementation**

|  |  |  |
| --- | --- | --- |
| **Table name** | **Attributes (Primary Keys & Foreign Keys)** | **Data type** |
| **papers** | PK: paper\_id  title  FK: target \_id | INTEGER NOT NULL  VARCHAR(255) NOT NULL  INTEGER NOT NULL |
| **depts** | PK: dept\_id  dept\_name | INTEGER NOT NULL  VARCHAR(20) NOT NULL |
| **faculty\_papers** | PK + FK: faculty\_id  PK + FK: paper\_id  FK: role\_id | INTEGER NOT NULL  INTEGER NOT NULL  INTEGER NOT NULL |
| **targets** | PK: target\_id  target\_name  FK: target\_type\_id  tier | INTEGER NOT NULL  VARCHAR(255) NOT NULL  INTEGER NOT NULL  INTEGER NULL |
| **events** | PK: event\_id  event\_date  FK: event\_type\_id  FK: paper\_id | INTEGER NOT NULL  DATE NOT NULL  INTEGER NOT NULL  INTEGER NOT NULL |
| **event\_types** | PK: event\_type\_id  event\_type\_name | INTEGER NOT NULL  VARCHAR (20) NOT NULL |
| **target\_types** | PK: target\_type\_id  target\_name | INTEGER NOT NULL  VARCHAR(255) NOT NULL |
| **roles** | PK: role\_id  role\_type\_name | INTEGER NOT NULL  VARCHAR(20) NOT NULL |
| **paper\_coauthors** | PK + FK: paper\_id  PK: coauthor\_name | INTEGER NOT NULL  VARCHAR(50) NOT NULL |
| **faculty** | PK: faculty\_id  faculty\_name | INTEGER NOT NULL  VARCHAR(50) NOT NULL |
| **faculty\_dept** | PK + FK: faculty\_id  PK + FK: dept\_id | INTEGER NOT NULL  INTEGER NOT NULL |

### Table 1 - Table name, Column with Primary key and Foreign Key, and Data Type

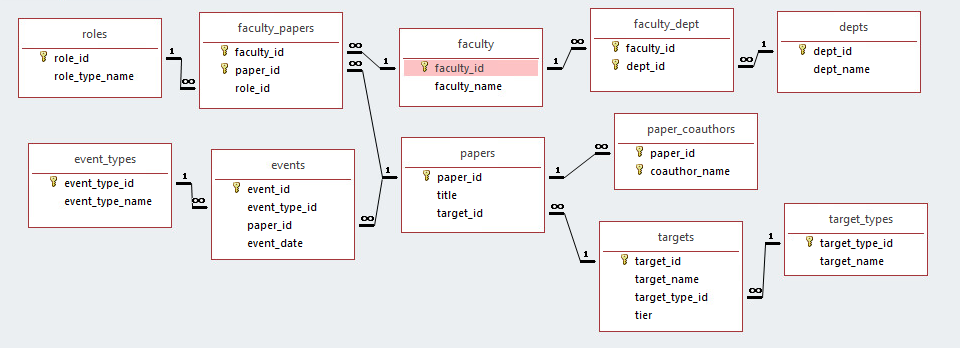
### 

All data except the *tier* in the *targets* table is NOT NULL, meaning the cell cannot be left empty and some data value must be entered. In contrast, not all targets have tiers, so the *tier* attribute can be NULL.

In terms of *id* data including *paper\_id, target\_id, dept\_id, faculty\_id, target\_type\_id, event\_id, event\_type\_id,* and *role\_id*, we choose INTEGER because it is a number that uniquely identifies those attributes.

In the *event* table, we use the data type for *event\_date* as DATE NOT NULL because the data put in the Excel file is date.

We choose VARCHAR(255) for long names such as *target\_name* and *title*. The field *dept\_name*, *faculty\_name* and *coauthor\_name* are used VARCHAR(50)as these are estimated to have less than 50 characters. We put VARCHAR(20) for *event\_type\_name* and *role\_type\_name* because these are not longer than 20 characters.



### Figure 1 - ENTITY RELATIONSHIP DIAGRAM

The eleven tables in the database including primary and referential constraints are the result of the SQL script in *Appendix 1***.**

## Description of Tables

***Faculty\_papers***is used to connect table *faculty, papers* and *roles* and it shows what papers are generated by which faculty and role. The columns are *faculty\_id, paper\_id* and *role\_id.*

* Primary Key: The combination of *faculty\_id* and *paper\_id* is the composite primary key. This combination is unique for each record in the table to retrieve the information of the faculty, paper and role.
* Foreign Keys: *faculty\_id, paper\_id, role\_id*
  + *Faculty\_id* and *paper\_id* are the primary keys of the *faculty* and *papers* table. These are referenced as the foreign keys in this table to connect the specific paper and faculty to the related role ID.
  + *Role\_id* is the primary key of the *roles* table. It is referenced as the foreign keys in this table to connect with each combination of faculty and paper to show the role ID. With this ID, we can see the name of role type in the *roles* table because of the connection.

***Targets***is an object table that holds data in terms of targets. The columns are *target\_id, target\_name, target\_type\_id,* and *tier.* The table is used to indicate a specific target with name, type and tier.

* Primary Key: *Target\_id* is the primary key of the *targets* table. This key separates each target to avoid confusion about differences between targets.
* Foreign Key: *Target\_type\_id* is the primary key of the target\_types table. It is referenced in this table to indicate the type ID of the target. By this ID, we can see the type of target in the target\_type\_id because of the connection between two tables.

***Papers*** is an object table that holds data about all the papers submitted by the University’s faculty. This table holds attributes such as *paper\_id, title* and *target\_id.* *Title* stores the title name of the paper. *Target\_id* is a foreign key on the *papers* table. It references the *targets* table which stores the data about the papers’ targets.

* Primary Key: paper\_id is the ID given to each paper. This allows us to uniquely identify each paper and avoid incorrect reference or duplicates in the future.

***Depts*** is a category table that holds the different departments at the University. This table’s purpose is to categorize the University’s departments. The columns on this table are *dept\_id* and *dept\_name*. *Dept\_name* is the representation of a department through short text.

* Primary key: *dept\_id* is the individual ID of a specific department. This identification connects a certain faculty member to a certain department. More than one *faculty\_id* can be connected to more than one *dept\_id*. This many-to-many relationship creates the need for the *faculty\_dept* table which is explained shortly after.

***Events*** is a table holding all the information regarding each unique event that occurred. The different columns are *Event\_ID*, *Event\_date*, Event\_Type\_ID and *Paper\_ID*.

* Primary Key: *Event\_ID* assigns an integer representation, ID number, for each event. Classifying events this way eliminates confusion of which event is being referenced if there are duplicates in the system.

**Event\_Types** is an object table that holds the data about all of the types of events. The columns include Event\_Type\_ID and Event\_Type\_Name. The Event\_Type\_ID column is significant because it groups similar events together in a group and helps reference these groups of events in the Events table.

* Primary Key: Event\_Type\_ID is the ID number of each unique event type. This key separates each event type by its unique ID number to avoid confusion between how different events are

***Target\_Types*** is an object table that holds data about all the different types of targets. The columns contained within the table are *target\_type\_id* and *target\_name*. The *target\_types* table itself has no foreign key restriction, however its primary key *target\_type\_id* serves as a foreign key to another table; *target\_type\_id* on the *targets* table.

* Primary Key: *target\_type\_id* is the primary key of the *target\_types* table and designates a unique integer to correspond with each type of event we want to create. This primary key identifies all the targets we wish to create which can then be checked against other tables.

***Roles*** is a category table identifying all the possible roles an individual in the database may hold. The columns contained within the table are *role\_id* and *role\_type\_name*. There are three different roles available: Lead, Co-lead, and Contributor. The *roles* table itself has no foreign key restriction, however its primary key *role\_id* serves as a foreign key to another table; *role\_id* on the *faculty\_papers* table.

* Primary Key: *role\_id* is the primary key of the *roles* table and designates a unique integer to correspond with each of the three roles. This primary key identifies each of the roles we wish to create which can then be checked against other tables.

***Faculty\_Dept*** is an intersection table holding all the information regarding which faculty are in which departments.The different columns are *Faculty\_ID* and Dept\_ID.

* Primary Key: Faculty\_ID assigns an integer representation, ID number, for each event. Classifying events this way eliminates confusion of which event is being referenced if there are duplicates in the system.
* Foregin Key: Dept\_ID assigns an integer representation, ID number, for each department. An important note is that this is a composite primary key meaning that Dept\_ID is a foregin key in another table. It is listed as a foreign key because Access limits each table to one primary key.

***Paper\_coauthors*** is an object table that holds the data about paper coauthors.

* Primary key: paper\_ID
* Foreign key: coauthor\_name

***Faculty*** is an object table holding all the information regarding faculty and their names. Each faculty member is assigned a unique faculty ID in case two or more faculty members have the same name.

* Primary Key: Faculty\_ID assigns an integer representation, ID number, for each faculty member.

### 

### 

### 

### 

### 

### 

### 

### 

### 

### 

# 

# 

# 

# **Appendix 1: SQL Script for Schema Generation**

CREATE TABLE target\_types

(target\_type\_id INTEGER NOT NULL,

target\_name VARCHAR(255) NOT NULL,

PRIMARY KEY (target\_type\_id));

CREATE TABLE roles

(role\_id INTEGER NOT NULL,

role\_type\_name VARCHAR(20) NOT NULL,

PRIMARY KEY (role\_id));

CREATE TABLE event\_types

(event\_type\_id INTEGER NOT NULL,

event\_type\_name VARCHAR (20) NOT NULL,

PRIMARY KEY (event\_type\_id));

CREATE TABLE faculty

(faculty\_id INTEGER NOT NULL,

faculty\_name VARCHAR(50) NOT NULL,

PRIMARY KEY (faculty\_id));

CREATE TABLE depts

(dept\_id INTEGER NOT NULL,

dept\_name VARCHAR(20) NOT NULL,

PRIMARY KEY (dept\_id));

CREATE TABLE targets

(target\_id INTEGER NOT NULL,

target\_name VARCHAR(255) NOT NULL,

target\_type\_id INTEGER NOT NULL,

tier INTEGER NULL,

PRIMARY KEY (target\_id),

FOREIGN KEY (target\_type\_id) REFERENCES target\_types (target\_type\_id));

CREATE TABLE papers

(paper\_id INTEGER NOT NULL,

title VARCHAR(255) NOT NULL,

target\_id INTEGER NOT NULL,

PRIMARY KEY (paper\_id),

FOREIGN KEY (target\_id) REFERENCES targets (target\_id));

CREATE TABLE faculty\_dept

(faculty\_id INTEGER NOT NULL,

dept\_id INTEGER NOT NULL,

PRIMARY KEY (faculty\_id, dept\_id),

FOREIGN KEY (faculty\_id) REFERENCES faculty (faculty\_id),

FOREIGN KEY (dept\_id) REFERENCES depts (dept\_id));

CREATE TABLE faculty\_papers

(faculty\_id INTEGER NOT NULL,

paper\_id INTEGER NOT NULL,

role\_id INTEGER NOT NULL,

PRIMARY KEY (faculty\_id, paper\_id),

FOREIGN KEY (faculty\_id) REFERENCES faculty (faculty\_id),

FOREIGN KEY (paper\_id) REFERENCES papers (paper\_id),

FOREIGN KEY (role\_id) REFERENCES roles (role\_id));

CREATE TABLE paper\_coauthors

(paper\_id INTEGER NOT NULL,

coauthor\_name VARCHAR(50) NOT NULL,

PRIMARY KEY (paper\_id, coauthor\_name),

FOREIGN KEY (paper\_id) REFERENCES papers (paper\_id));

CREATE TABLE events

(event\_id INTEGER NOT NULL,

event\_type\_id INTEGER NOT NULL,

paper\_id INTEGER NOT NULL,

event\_date DATE NOT NULL,

PRIMARY KEY (event\_id),

FOREIGN KEY (event\_type\_id) references event\_types(event\_type\_id),

FOREIGN KEY (paper\_id) REFERENCES papers (paper\_id));