

Catalyst Maria High School – Class Grid

Melissa Mosier

IST 659 Data Administration Concepts and Management

8/17/2022

**Summary**

Catalyst Maria High School is creating a database for their class grid. This is how they will assign teachers and students to their classes for the coming semester. We will not get into Human Resources, teacher and admin relationships, additional staff, materials for classes, etc.; this is not meant to be a full mapping of relationships between everyone in this system. It is only to organize class assignments.

Teachers and administration from the same department will work together to create courses for their students. They will create a number of courses for each department. Each course will have at least one section but may have more. Each section will have at least one teacher and many students.

Stakeholders: Administrators, Teachers, Students

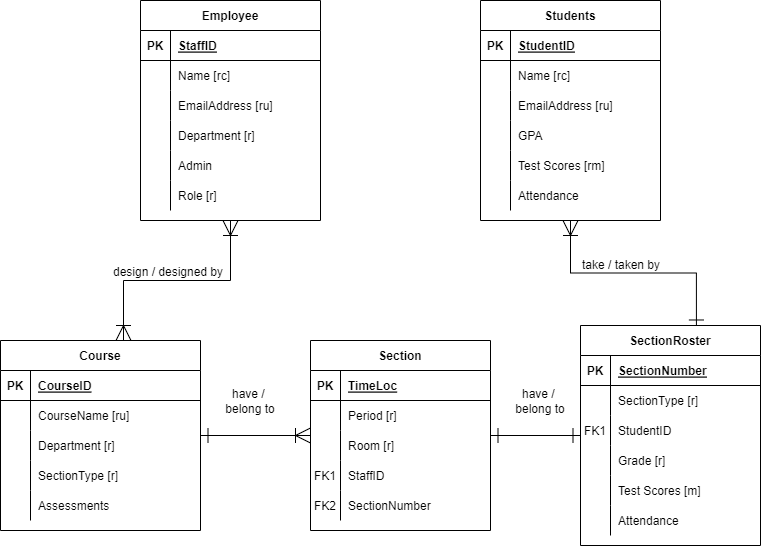
Business Rules:

* Many employees can design a course (this will usually include at least one teacher and one administrator). Many courses can be designed by an employee. Generally employees will design courses according to which department they belong.
* A course may have many sections. A section belongs to one course.
* Each section will have its own roster. A roster belongs to one section. Administrative information for each section (where it is, during which period, who teaches it) is kept separately from teacher-student information (section – DI, GENED, AP, etc.; grades; etc.)
* Many students will go on one roster (many students will take one section of a class together). A roster will only have a list of students for one class section (that class section will be taken by one roster of students).

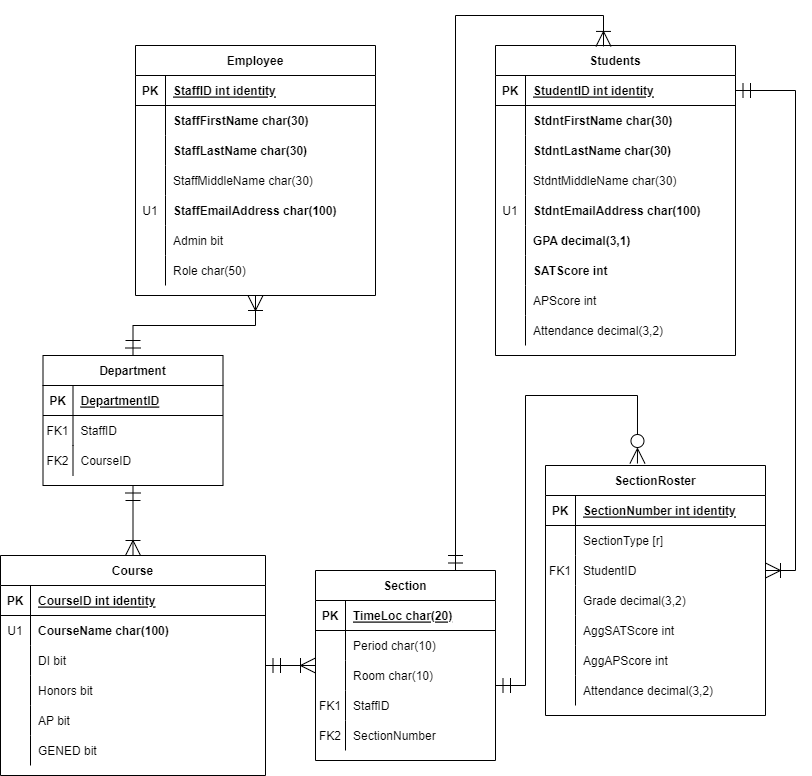
Data Questions:

* Can we pull Teacher and Student schedules on request?
* What are the metrics of students in each Course Section? (Grades, GPA, Standardized Tests, Attendance)
* Can we visualize grade trends in each Course Section? Compare between Teachers and Section Types of the same Course?

**Conceptual Model**



**Normalized Logical Model**



Additional Notes / Glossary:

Staff Table

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Comments / Examples** |
| **StaffID** | **Int identity** | **Primary Key; randomly generated** |
| **StaffFirstName** | **Char(25)** |  |
| **StaffLastName** | **Char(25)** |  |
| StaffMiddleName | Char(25) |  |
| **StaffEmailAddress** | **Char(100)** | **Unique** |
| Admin | Bit | Ex: 0 or 1 |
| Role | Char(50) | Ex: “Algebra 2 Teacher” vs “Math Coach” (Coach is an admin role) |

Course Table

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Comments / Examples** |
| **CourseID** | **Int identity** | **Primary Key; randomly generated** |
| **CourseName** | **Char(100)** | **Unique. Ex: “Psychology”** |
| DI | Bit | DI = Direct Instruction |
| Honors | Bit |  |
| AP | Bit | AP = Advanced Placement |
| GENED | Bit | GENED = General Education |

Notes:

* A “1” for DI / Honors / AP / GENED indicates whether we are required to offer this type of section for this course. Later, we will indicate if a particular section is DI / Honors / AP / GENED as SectionType.

Student Table

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Comments / Examples** |
| **StudentID** | **Int identity** | **Primary Key; randomly generated** |
| **StdntFirstName** | **Char(25)** |  |
| **StdntLastName** | **Char(25)** |  |
| StdntMiddleName | Char(25) |  |
| **StdntEmailAddress** | **Char(100)** |  |
| **GPA** | **Decimal (3,1)** | **Ex: “3.24”** |
| **SATScore** | **Int** |  |
| APScore | Int |  |
| Attendance | Decimal (3,2) | This is a percentage stored as a decimal. Ex: “98.2” |

Roster Table

|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Comments / Examples** |
| **SectionNumber** | **Int identity** | **Primary Key; randomly generated** |
| **SectionType** | **Char(25)** | **Categorical. Ex: “GENED”** |
| StudentID | Foreign Key | From Student table |
| Grade | Decimal (3,2) |  |
| AggSATScore | Int | Aggregated SAT Score |
| AggAPScore | Int | Aggregated AP Score |
| Attendance | Decimal (3,2) |  |

* This is a massive table that could be split into many smaller tables if each SectionNumber has its own table.
  + Section Number and SectionType would repeat while each StudentID gets their own row with their class Grade. This is a master roster grid.
  + In 3NF, each SectionNumber has its own table, and each StudentID still gets their own row. This is the traditional class roster as given to teachers.

Section Table

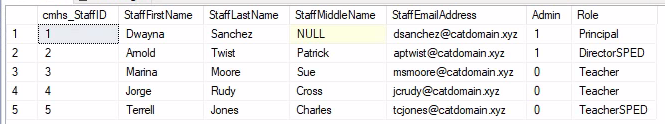
|  |  |  |
| --- | --- | --- |
| **Field Name** | **Data Type** | **Comments / Examples** |
| **TimeLoc** | **Char(20)** | **Primary Key. Derived from Period and Room.** |
| **Period** | **Char(10)** |  |
| **Room** | **Char(10)** |  |
| StaffID | Foreign Key | From Employee table |
| SectionNumber | Foreign Key | From SectionRoster table |

Notes:

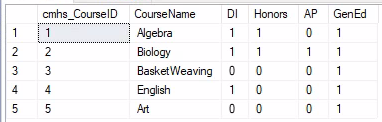
* No two classes can be taught in the same room during the same period. This PK (TimeLoc) could be the same as the SectionNumber PK, but this information is administration-facing, whereas SectionRoster is student-teacher-facing.

**Screenshots**

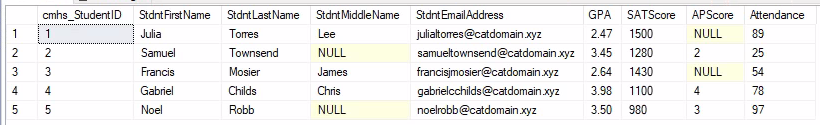
Staff Table



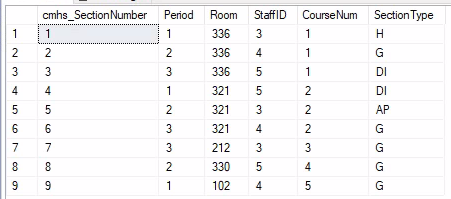
Course Table



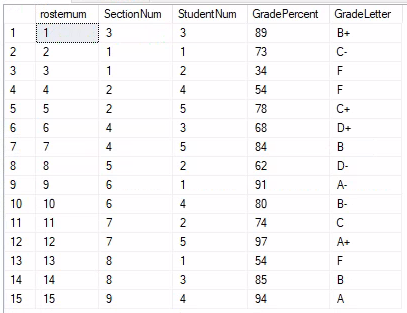
Student Table



Section Table

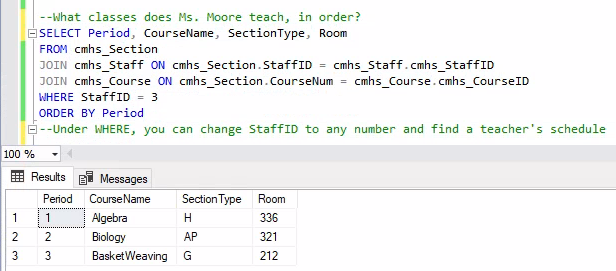


Roster Table

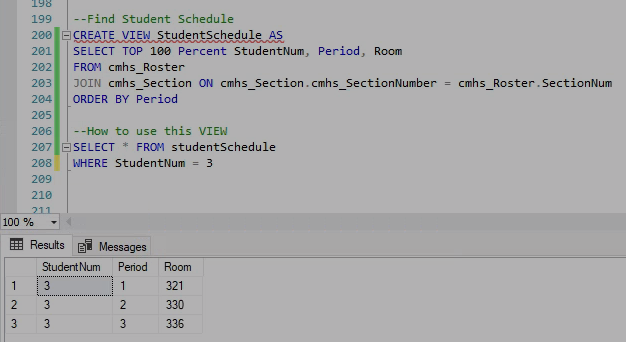


Answering Data Questions:

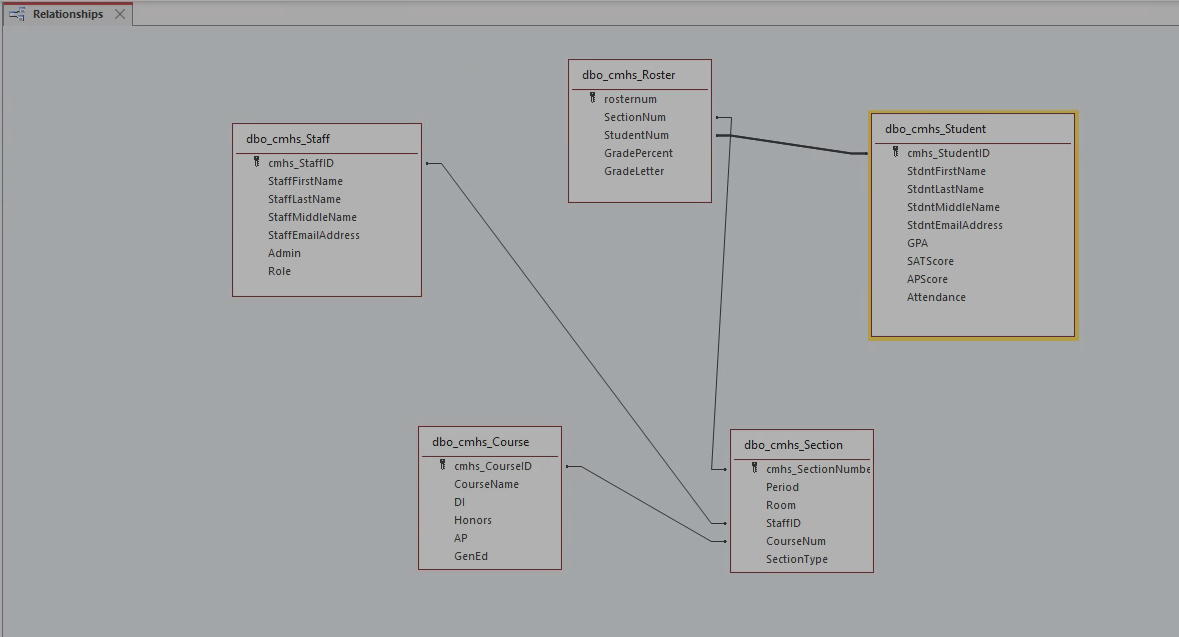
* SQL Code to Pull Teacher Schedule



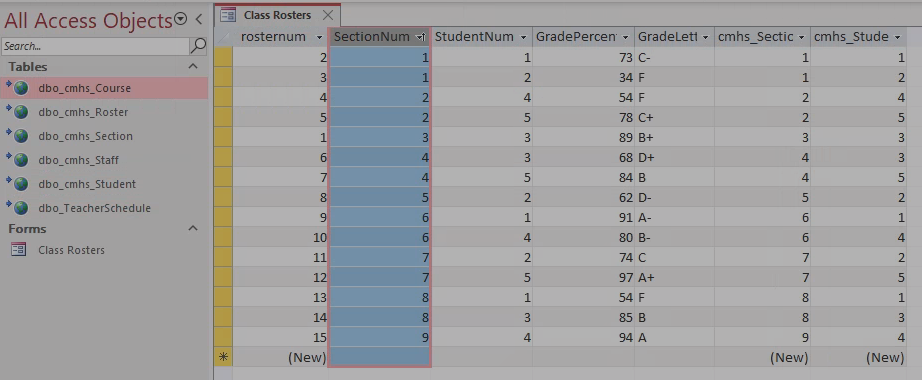
* Pull Student Schedule



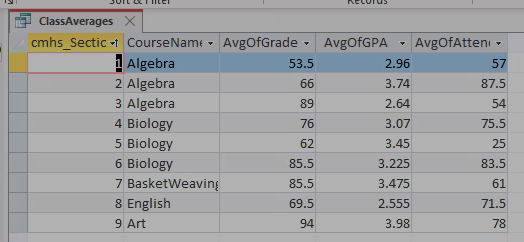
Implementation on Access



Sorting in tables



* Aggregating metrics for each Course Section (Grades, GPA, Standardized Tests, Attendance)



We are able to answer all data questions!

**Conclusion**

I did not anticipate, when creating this database, that there would be so much reiteration in the process. At every step I found that there were modifications to be made in order for this database to function well.

* In creating the conceptual model, I conferenced with a high school Dean of Students and a database manager who uses Salesforce. They gave me feedback on content (like including SectionType, which is important for schools to retain their accreditation) and form (like figuring out how to separate and organize Sections and Section Rosters). It would turn out later that these were some the hardest parts to conceptualize for this database. It was easy to identify what information we needed to store for Staff and Students. Designing the relationships between Staff and Students, especially to their Courses, was difficult.
* In transition from the conceptual to logical models, I separated Department. It slipped my mind in transition into SQL code, since my focus fell on Student-Teacher relationships through Courses rather than their relationships to Administration. I realized it was less a structural necessity and more a referential label. I did not bother to built it into the final database, though I am sure that if a school would ask for it I were ever to implement this.
* When finally transitioning into SQL code, the Staff, Student and Course tables experienced almost no structural changes. However,
  + The Section Table needed to have SectionNumber as its Primary Key instead of that being a Key on the Roster, so that it could actually identify different classes and be referenced on the Roster when assigning students to those classes. SectionType and CourseID also needed to be in this table. I realized that this was the table that would link Courses to Staff/Teachers, whereas…
  + The Roster would link Courses to Students. AggSATScore, AggAPScore, and Attendance were all aggregated/derived attributes, so they didn’t need to be included here. Rather, we only needed to gather SectionNumbers, StudentIDs, and Grades here.
* I also converted Attendance and Grades to integers because I did not want to deal with more decimals. I literally just did not feel like it. This was honestly just a modeling choice; if I were ever to implement this with a school it would most likely need to be a percent.

This assignment does meet expectations for creating and storing a class grid (referring to Business Rules.

* Each SectionNumber is associated with a StaffID.
* Many SectionNumbers may be associated with a CourseID.
* A Section table has administrative information, like where a class is, when it takes place, who teaches it, what type of class, etc. A Roster table has Student and Grade information.
* From the Roster table, we can have both a master grid table that tells us which students take which Section of a class, and we can also pull traditional rosters for just one Section with all its students.