Julio Claudian Family Tree

Conor O'Riordan

Professor Thomas McKee

Database Design

5 December, 2019

INTRODUCTION

The late Roman Republic and the early Principate are the two most well-known eras from antiquity. From Cicero to Cleopatra, the most lauded and infamous individuals lived during the turn of the millennium - between the first century BC and first century AD.

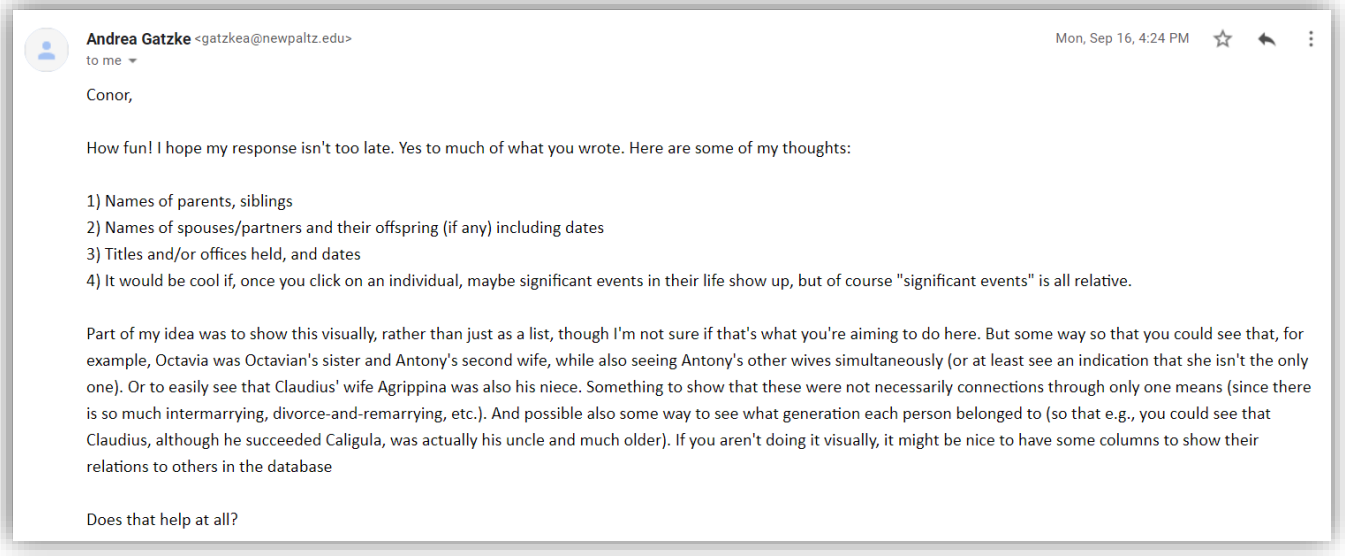
As Shakespeare once wrote, "All the world's a stage, And all the men and women merely players." If that is the case, the Julio-Claudian family showcased the most intricate cast in history, for these family members and their allies set into motion the fall of the republic and its rise as an empire. Even Shakespeare could not resist such a compelling narrative and used the fall of the Roman Republic as the setting to his *Julius Caesar* production.

However, due to Roman social customs and an insatiable lust for power, the Julio-Claudian family is not a simple family tree. Members killed off competing branches, man and women married multiply times to cement alliances, parents exiled their own children, and barren members adopted children to preserve their legacy. The complexity of these familial relations poses a steep learning curve to the uninitiated who wish to learn more about this era.

This Julio-Claudian family tree database is meant to be an educational tool for all skill levels. It assumes very little prior knowledge of Roman history and customs, so students and the general public can quickly learn through simple queries. Seasoned historians and professors are not forgotten; though this database does provide tables on Julio-Claudian members and their relationships such as marriages and personal information, it also provides a timeline of their time in public office, involvement in famous events, and their depictions on famous artifacts, statuary, monuments, and paintings. No matter the skill level, this database will be invaluable to expert and layperson alike.

CONCEPTION

Professor Andrea Gatzke, one of my former professors from the New Paltz history department, inspired this topic. When I first told her that I was pursuing a master's degree in computer science, she confessed that she always wanted an interactive model to help her students understand the intricacies of the Julio-Claudians. Now, her grand scheme is beyond the scope of this database (such as a front-end interactive app or website), but Professor Gatzke provided many of the business rules and constraints for this project. Below is a snippet of our email exchange about the topic.



Andrea Gatzke <gatzkea@newpaltz.edu> Mon, Sep 16, 4:24 PM

to me

Conor,

How fun! I hope my response isn't too late. Yes to much of what you wrote. Here are some of my thoughts:

1) Names of parents, siblings
2) Names of spouses/partners and their offspring (if any) including dates
3) Titles and/or offices held, and dates
4) It would be cool if, once you click on an individual, maybe significant events in their life show up, but of course "significant events" is all relative.

Part of my idea was to show this visually, rather than just as a list, though I'm not sure if that's what you're aiming to do here. But some way so that you could see that, for example, Octavia was Octavian's sister and Antony's second wife, while also seeing Antony's other wives simultaneously (or at least see an indication that she isn't the only one). Or to easily see that Claudius' wife Agrippina was also his niece. Something to show that these were not necessarily connections through only one means (since there is so much intermarrying, divorce-and-remarrying, etc.). And possible also some way to see what generation each person belonged to (so that e.g., you could see that Claudius, although he succeeded Caligula, was actually his uncle and much older). If you aren't doing it visually, it might be nice to have some columns to show their relations to others in the database

Does that help at all?

I chose MySQL for my DBMS specifically for her future hopes of implementing a front-end website or app mentioned above. Lucid Charts was recommended by a fellow student who took Database Design in the past; I found the modelling software easy to understand and convenient, albeit a bit frustrating when designing and implementing supertypes and subtypes.

ERD & NORMALIZATION

The next few pages will show the progression of the ERD throughout the semester. Coming into the class with no background in computer science or database design, my models changed drastically with time. With each new chapter and topic, my database evolved to reflect those new tools and constraints. On the following models, I have included "sticky notes" to describe my thought process and decision making. Some major changes to entities, relations, and datatypes near the end of the project will be discussed in their own section.

INSERTING DATA & LATE-TERM MODIFICATIONS

```
CREATE DATABASE Julio_Claudian_Family_Tree;

CREATE Table person_t (
        PersonID        Integer         NOT NULL            AUTO_INCREMENT,
        KnownName       varchar(30)     NOT NULL,
        Praenomen       varchar(15)     SET DEFAULT 'N/A',
        Nomen           varchar(15),
        Cognomen        varchar(15),
        AdditionalCognomen varchar(15)  SET DEFAULT 'N/A',
        Father          varchar(30)     SET DEFAULT 'Unknown',
        FatherID        Integer         NOT NULL,
        Mother          varchar(30)     SET DEFAULT 'Unknown',
        MotherID        Integer         NOT NULL,
        DOB             Integer,
        DOD             Integer,
        Julian          bool            NOT NULL,
        Claudian        bool            NOT NULL,
        Barren          bool            NOT NULL,
CONSTRAINT person_pk PRIMARY KEY (PersonID),
CONSTRAINT person_fk1 FOREIGN KEY (FatherID) REFERENCES person_t (PersonID),
CONSTRAINT person_fk2 FOREIGN KEY (MotherID) REFERENCES person_t (Person_ID));

INSERT INTO person_t(PersonID, KnownName, Praenomen, Nomen, Cognomen, Father,
FatherID, Mother, MotherID, DOB, DOD, Julian, Claudian, Barren) VALUES (14, 'Julius
Caesar', 'Gaius', 'Julius', 'Caesar', 'Gaius Julius Caesar III', 6, 'Aurelia Cotta', 12, -100, -44, 1,
0, 0);

INSERT INTO person_t(PersonID, KnownName, Praenomen, Nomen, Cognomen,
AdditionalCognomen, Father, FatherID, Mother, MotherID, AdoptiveFather, DOB, DOD, Julian,
Claudian, Barren) VALUES (37, 'Tiberius', 'Tiberius', 'Julius', 'Caesar', 'Claudius', 'Tiberius
Claudius Nero', 32, 'Livia Drusilla', 34, 'Augustus', -16, 37, 1, 1, 0);

INSERT INTO person_t (PersonID, KnownName, Praenomen, Nomen, Cognomen, FatherID,
MotherID, Julian, Claudian, Barren) VALUES ('Sextus Julius Caesar I', 'Sextus', 'Julius',
'Caesar', 999, 999, 1, 0, 0);

INSERT INTO person_t (PersonID, KnownName, FatherID, MotherID, Julian, Claudian,
Barren) VALUES ('Unknown', 999, 999, 1, 0, 0);
```

```
CREATE TABLE dynasty_t (
        PersonID        Integer         NOT NULL,
        Julius          bool,
        Augutus         bool,
        Tiberius        bool,
        Caligula        bool,
        Claudius        bool,
        Nero            bool,
CONSTRAINT dynasty_pk PRIMARY KEY (PersonID),
CONSTRAINT dynasty_fk FOREIGN KEY (PersonID) REFERENCES person_t (PersonID));

INSERT INTO dynasty_t(PersonID, Julius) VALUES (14, 1), (31, 1, 1), (37, 1, 1), (52, 0, 1, 1,
1, 1, 0);
```

```
CREATE TABLE adoptionlist_t (
        FatherID                Integer         NOT NULL,
        ChildID                 Integer         NOT NULL,
        YearOfAdoption          Integer,
CONSTRAINT adoptionlist_pk PRIMARY KEY (FatherID, ChildID),
CONSTRAINT adoptionlist_fk1 FOREIGN KEY (FatherID) REFERENCES person_t
(PersonID),
CONSTRAINT adoptionlist_fk2 FOREIGN KEY (ChildID) REFERENCES person_t
(PersonID));

INSERT INTO adoptionlist_t VALUES (14, 31, -44), (31, 37, 4), (31, 47, 4), (31, 48, -17), (31,
49, -17), (37, 51, 4);
```

```
CREATE TABLE marriagelist_t (
        HusbandID               Integer         NOT NULL,
        WifeID                  Integer         NOT NULL,
        NumberOfChildren        Integer         NOT NULL,
        YearOfMarriage          Integer,
        YearOfSeperation        Integer,
CONSTRAINT marriagelist_pk PRIMARY KEY (HusbandID, WifeID),
CONSTRAINT marriagelist_fk1 FOREIGN KEY (HusbandID) REFERENCES person_t
(PersonID),
CONSTRAINT marriagelist_fk2 FOREIGN KEY (WifeID) REFERENCES person_t
(PersonID));

INSERT INTO marriagelist_t VALUES (14, 21, 1, -84, -69, 'Her death'), (14, 20, 0, -67, -61,
'Divorce'), (14, 19, 0, -59, -44, 'His death'), (31, 33, 1, -40, -38, 'Divorce'), (31, 34, 0, -37, 14,
'His death');
```

```
CREATE TABLE governmentoffice_t (
        OfficeID            Integer             NOT NULL,
        OfficeName          varchar(20)         NOT NULL,
CONSTRAINT governmentoffice_pk PRIMARY KEY (OfficeID));

INSERT INTO governmentoffice_t VALUES (1, 'Military Service'), (2, 'Military Tribune'), (3,
'Tribune of the Plebs'), (4, 'Aedile');
```

```
CREATE TABLE resume_t (
        OfficeID            Integer             NOT NULL,
        PersonID            Integer             NOT NULL,
        StartOfOffice       Integer             NOT NULL,
CONSTRAINT resume_pk PRIMARY KEY (OfficeID, PersonID, StartOfOffice),
CONSTRAINT resume_fk1 FOREIGN KEY (OfficeID) REFERENCES governmentoffice_t
(OfficeID),
CONSTRAINT resume_fk2 FOREIGN KEY (PersonID) REFERENCES person_t (PersonID));

INSERT INTO resume_t VALUES (14, 6, -59), (14, 6, -48), (14, 6, -46), (14, 11, -49);
```

```
CREATE TABLE famousevent_t (
        EventName           Integer             NOT NULL,
        Year                Integer             NOT NULL,
        Description         varchar(100),
CONSTRAINT famousevent_PK PRIMARY KEY (EventName));

INSERT INTO famousevent_t VALUES ('Battle of Teutoburg Forest', 9, 'The Roman
commander Varus and three legions are slaughtered in an ambush led by Arminius'), ('Battle of
Actium', -31, 'Decisive victory for Octavian against Mark Anthony'), ('Assassination of Julius
Caesar', -44, 'Julius Caesar is assassinated by several senators'), ('Julius Caesar is declared
Dictator Perpetuus', -44, 'Julius Caesar is declared dictator for life');
```

```
CREATE TABLE eventrole_t (
        EventName           varchar(50)         NOT NULL,
        PersonID            Integer             NOT NULL,
        AllyID              Integer,
        RoleInEvent         varchar(50),

CONSTRAINT eventrole_pk PRIMARY KEY (EventName, PersonID),
CONSTRAINT eventrole_fk1 FOREIGN KEY (EventName) REFERENCES famousevent_t
(EventName),
CONSTRAINT eventrole_fk2 FOREIGN KEY (PersonID) REFERENCES person_t (PersonID)
CONSTRAINT eventrole_fk3 FOREIGN KEY (AllyID) REFERENCES ally_t (AllyID));
```

```
INSERT INTO eventrole_t VALUES ('Assassination of Julius Caesar', 14, 1, 'Assassinated'),
('Battle of Actium', 31, 'Victorious at Actium'), ('Battle of Actium', 28, 3, 'Defeated at Actium'),
('Battle of Actium', 41, 'General for Octavian');
```

```
CREATE TABLE visual_t (
        ArtID           Integer         NOT NULL    AUTO_INCREMENT,
        TitleOfWork     varchar(35)     NOT NULL,
        Medium          varchar(15)     NOT NULL,
        Artist          varchar(30),
        Museum          varchar(50),
        Description     varchar(100),
CONSTRAINT visual_pk PRIMARY KEY (ArtID));

insert into visual_t (TitleOfWork, Medium, Museum, Description) values
('Augustus of Primaporta', 'White Marble', 'Vatican Museums, Rome', 'A full-length statue of
Augustus');

insert into visual_t values (2, 'The Death of Caesar', 'Oil on Canvas', 'Jean-Leon Gerome',
'Walters Art Museum, Baltimore', 'Painting depicting the conspirators leaving the Theatre of
Pompey');

insert into visual_t(TitleOfWork, Medium, Museum, Description) values
('Bust of Emperor Claudius', 'White Marble', 'Naples National Archaeological Museum, Naples',
'A bust of Claudius while emperor'), (''Marc Athony's Oration at Caesar's Funeral'', 'Oil on
Canvas', 'Hartlepool Museums and Heritage Service, Hartlepool', 'Mark Anthony delivering his
iconic oration made famous by Shakespeare');
```

```
CREATE TABLE visuallist_t (
        ArtID           Integer         NOT NULL,
        PersonID        Integer         NOT NULL,
CONSTRAINT visuallist_pk PRIMARY KEY (ArtID, PersonID),
CONSTRAINT visuallist_fk1 FOREIGN KEY (ArtID) REFERENCES visual_t (ArtID),
CONSTRAINT visuallist_fk2 FOREIGN KEY (PersonID) REFERENCES person_t
(PersonID));

INSERT INTO visuallist_t VALUES (1, 31), (2, 14), (3, 52), (4, 14), (4, 28);
```

```
CREATE TABLE allylist_t (
        AllyID              Integer         NOT NULL,
        PersonID            Integer         NOT NULL,
        StartOfAlliance     Integer,
        EndOfAlliance       Integer,
CONSTRAINT allylist_pk PRIMARY KEY (AllyID, PersonID),
CONSTRAINT allylist_fk1 FOREIGN KEY (AllyID) REFERENCES ally_t (AllyID),
CONSTRAINT allylist_fk2 FOREIGN KEY (PersonID) REFERENCES person_t (PersonID));

INSERT INTO allylist_t VALUES (1, 14, -48, -44), (2, 14, -60, -53), (3, 14, -41, -30),
(2, 25, -48, 44);
```

```
CREATE TABLE ally_t (
        AllyID              Integer         NOT NULL,
        KnownName           varchar(30)     NOT NULL,
        DOB                 Integer,
        DOD                 Integer,
CONSTRAINT ally_pk PRIMARY KEY (AllyID);

INSERT INTO ally_t VALUES (1, 'Brutus', -85, -42), (2, 'Crassus', -112, -53), (3, 'Cleopatra', -
69, -30), (4, 'Varus', -46, 9);
```

I discovered more business rules when I began entering data and encountered warning messages due to primary key constraints and referential integrity. The most glaring error occurring when entering data for governmentoffice_t and resume_t. In Roman society, many Romans held one position for one term before applying for the next – they were always climbing up the social ladder with a clear progression. However, Romans could serve as consul, a position like the presidency, for two terms. Originally, the primary key for resume_t was a composite of PersonID and OfficeID, but this caused duplicate primary keys for people who served as consul twice. It was a glaring error, but I only caught it thanks to placing constraints. I then updated the composite key to include StartOfOffice to differentiate multiple terms in office.

Another late-game modification was the datatype for the chronological attributes. Apparently, the DATE datatype allows you to use both BC/AD notation. However, due to the lack of concrete dates from the historical sources, I chose VARCHAR because it was easier to implement. This led to two problems – naming conventions and writing queries. Common Era (BCE/CE) notation is phasing out the former BC/AC standard. That means inconsistencies may occur if multiple people with different preferences enter data into this database without being aware of the business rules. Though I prefer the BC/AC due to its Latin roots, it has its own quirks at well. For example, the proper way to write dates using this notation is 44 BC and AD 9. Once again, this business rule is prone to user error when inserting values. Also, writing queries using dates was extremely frustrating using multiple wild card notations to isolate BC/AD and the year of interest. Using the INTEGER datatype simplified queries, opened the possibility of aggregated functions, and removed ambiguity when inserting data into the tables caused by both BC/AD and BCE/CE.

SAMPLE QUERIES

Inner Join

QUESTION: List the alliance of all the Julio-Claudians. Include their IDs and names.

SELECT P.PersonID, P.KnownName, AL.PersonID, AL.AllyID, A.AllyID, A.KnownName
FROM person_t P, allylist_t AL, ally_t A
WHERE P.PersonID = AL.PersonID AND AL.AllyID = A.AllyID;
SELECT * FROM allylist_t;

PersonID	KnownName	PersonID	AllyID	AllyID	KnownName
14	Julius Caesar	14	1	1	Brutus
14	Julius Caesar	14	2	2	Crassus
14	Julius Caesar	14	3	3	Cleopatra
25	Pompey	25	2	2	Crassus

Outer Join

QUESTION: What offices did Julius Caesar hold in his lifetimes? Which offices did he not hold?

SELECT *
FROM resume_t R RIGHT JOIN
governmentoffice_t GO
ON (R.OfficeID = GO.OfficeID)
WHERE PersonID = 14 OR PersonID
IS NULL;

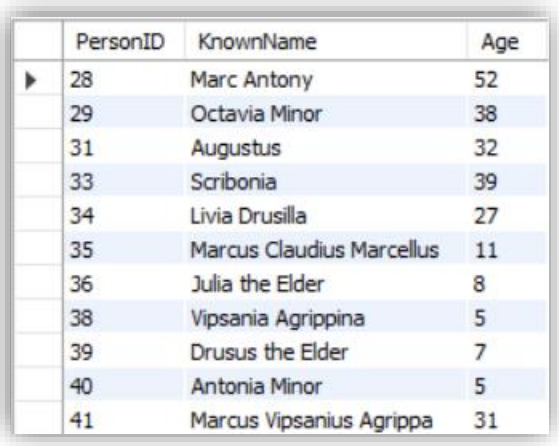
PersonID	OfficeID	StartOfOffice	OfficeID	OfficeName
NULL	NULL	NULL	1	Military Service
14	2	-73	2	Military Tribune
14	3	-69	3	Quaestor
14	4	-65	4	Aedile
14	5	-62	5	Praetor
14	6	-59	6	Consul
14	6	-48	6	Consul
14	6	-46	6	Consul
14	6	-44	6	Consul
14	7	-61	7	Governor
14	7	-58	7	Governor
NULL	NULL	NULL	8	Censor
NULL	NULL	NULL	10	Princeps Senatus
14	11	-49	11	Dictator
NULL	NULL	NULL	13	Emperor

Correlated Sub-Query

QUESTION: Which Julio-Claudians were alive during the Battle of Actium that took place in -31 BC?

How old where they when the battle occurred?

```
Select PersonID, KnownName, SUM(-31-DOB) AS Age
FROM person_t P
WHERE EXISTS
        (Select *
   FROM famousevent_t FE
   WHERE EventName = 'Battle of Actium' AND
FE.Year BETWEEN P.DOB AND P.DOD)
GROUP BY PersonID;
```

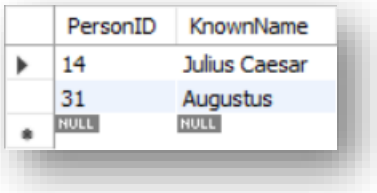


PersonID	KnownName	Age
28	Marc Antony	52
29	Octavia Minor	38
31	Augustus	32
33	Scribonia	39
34	Livia Drusilla	27
35	Marcus Claudius Marcellus	11
36	Julia the Elder	8
38	Vipsania Agrippina	5
39	Drusus the Elder	7
40	Antonia Minor	5
41	Marcus Vipsanius Agrippa	31

Sub-Query

QUESTION: List all the men who had a divorce.

```
Select PersonID, KnownName
FROM person_t
WHERE PersonID IN
        (SELECT HusbandID
        FROM marriagelist_t
        WHERE ReasonforSeperation = 'Divorce');
```



PersonID	KnownName
14	Julius Caesar
31	Augustus
NULL	NULL

Self Join

QUESTION: For a given person, who was his/her father and grandfathers? Please note that the ancestors of people who married into the family have default values of KnownName = 'Unknown' and PersonID = 999.

```
SELECT P1.PersonID, P1.KnownName AS REFERENCE,
        P2.PersonID, P2.KnownName AS FATHER,
        P3.PersonID, P3.KnownName AS PATERNAL_GRANDFATHER,
        P5.PersonID, P5.KnownName AS MATERNAL_GRANDFATHER
FROM person_t P1, person_t P2, person_t P3, person_t P4, person_t P5
WHERE P1.FatherID = P2.PersonID AND P2.FatherID = P3.PersonID
        AND P1.MotherID = P4.PersonID AND P4.FatherID = P5.PersonID;
```

PersonID	REFERENCE	PersonID	FATHER	PersonID	PATERNAL_GRANDFATHER	PersonID	MATERNAL_GRANDFATHER
41	Marcus Vipsanius Agrippa	999	Unknown	999	Unknown	999	Unknown
42	Livilla	39	Drusus the Elder	32	Tiberius Claudius Nero	28	Marc Antony
43	Drusus the Younger	37	Tiberius	32	Tiberius Claudius Nero	999	Unknown
44	Tiberius Gemellus	43	Drusus the Younger	37	Tiberius	39	Drusus the Elder
45	Julia Livia	43	Drusus the Younger	37	Tiberius	39	Drusus the Elder
46	Julia the Younger	41	Marcus Vipsanius Agri...	999	Unknown	31	Augustus
47	Agrippa Postumus	41	Marcus Vipsanius Agri...	999	Unknown	31	Augustus
48	Gaius Julius Caesar	41	Marcus Vipsanius Agri...	999	Unknown	31	Augustus
49	Lucius Caesar	41	Marcus Vipsanius Agri...	999	Unknown	31	Augustus
50	Agrippina the Elder	41	Marcus Vipsanius Agri...	999	Unknown	31	Augustus
51	Germanicus	39	Drusus the Elder	32	Tiberius Claudius Nero	28	Marc Antony
52	Claudius	39	Drusus the Elder	32	Tiberius Claudius Nero	28	Marc Antony

BONUS – MS ACCESS ODBC

Since I chose MySQL for my DBMS of choice, I decided upon establishing an ODBC with Microsoft Access; the spreadsheet nature of Access made it ideal for entering repetitive values, especially in person_t due to its many attributes. It was a rather simple process to establish the connection and to propagate MS Access with linked tables originating from MySQL. Being a Microsoft product, Access should be easier for my history teacher Andrea Gatzke to download and understand compared to MySQL which would have to be downloaded on her personal device. We could collaborate as student and professor once more using MS Access as the common link. She could insert and review the data for historical accuracy, and I could focus on writing SQL queries to common questions regarding the Julio-Claudians as I did above in my sample queries.