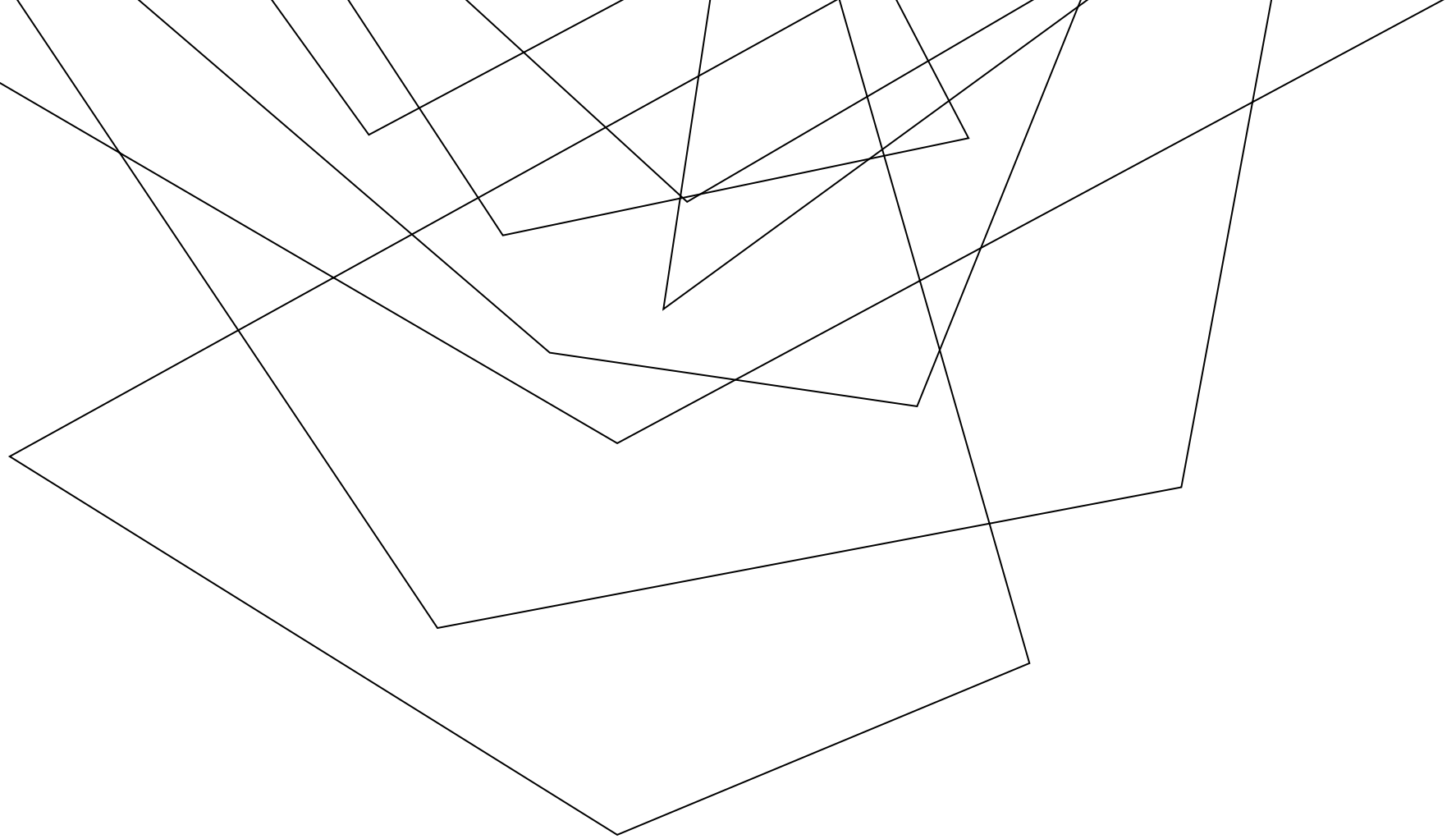


ALUMNI DATABASE

Alphonso Anderson

Celso Fuentes

Dennis Thomas

Narkwor Mensah

AGENDA

- Introduction
- Logical Aspects
- Physical Aspects
- Implementation
- Summary

INTRODUCTION

This presentation was created to show how our Alumni database is constructed and how it functions. It will provide a comprehensive description of what we want in a database. We will present what was determined to be the entities, their relations with each other, what are the attributes each entity holds and the entity constraints such as keys and referential integrity.

Once the relationships have been discussed, we will integrate our database model into SQL and add the required data and constraints to complete our database followed by giving examples of queries to show it works properly.

LOGICAL DESIGN

Knowledge → Information → Data

SELECTING ENTITIES

In this database, we needed to start by deciding what should be our entities.

The entities we've chosen were

Alumni, CSU_Students, Company, Funds, Alumni_Degree, Expertise, Events, Donations, AlumniEmployment

The entities chosen will provide a foundation for how the data will be inserted into the database

ENTITY SETS

Alumni	CSUStudents	Company	Funds	AlumniDegree
• The alumni will feature the Clayton State University Alumni who sign up to the program. Their name, major, email, phone and LinkedIn will be provided.	• For the Clayton State students interested in mentorship, they will provide their name, CSUID, type, phone and major.	• A particular number of companies that the alumni may work for. Its companyID, name, location and size is provided.	• Donations from alumni are sent to fund a certain department. The features a fund ID, department, lake connect, and total donations.	• A list of degrees earned by the alumni while attending. The degreeID and area name are provided.

ENTITY SETS

Expertise

- A field of expertise the students can browse and find whose matches their search. An expertiseID and name is provided.

Donations

- List of donations sent from the alumni. Every donation has a DonationID, AlumniID from alumni, donation total and fundID.

Events

- List of events provided to selected alumni. Provided is the EventID, name, location, date, start and end time.

AlumniEmployment

- This features a set of employment history by a single alumni. This should feature the alumniID, dateJoined, salary, dateLeft and jobTitle.

THE RELATIONSHIP OF THE ENTITIES

Now that the entities were chosen, we must determine what the relationships between the entities are.

We have determined the relationship are

Alumni mentor CSU Students	Alumni contributes donations	Alumni is employed at Company	Alumni earned a Degree	Alumni has expertise	Alumni is invited to events	Donations added to funds

PHYSICAL DESIGN

Of Database

CONSTRAINTS WITH THE ENTITY RELATIONS MODEL

- Now that the entities were chosen, we determined many of the entities would have either a primary key or a foreign key to preserve referential integrity.
- For example,
  - every alumni has a unique ID number for their membership. This will be their primary key.
  - For the Donation entity, there must be a reference to the parent entity. So, we added the foreign key alumniID from the Alumni entity to preserve referential integrity.

CONSTRAINTS WITH THE ENTITY RELATIONS MODEL

- We also determined what would be the datatype for each attribute so that any invalid insertion of data would be rejected.
- For example,
  - CompanyID is a numeric datatype that only uses three digits and nothing more

CONSTRAINTS WITH THE ENTITY RELATIONS MODEL

- When implementing the attribute datatypes, we also chose which would ones are not allowed to be null when adding data.
- For example,
  - If we were to add an alumni to the alumni entity without an ID, it would be rejected for violating the its entity integrity.

DATABASE CONSTRAINTS

```
ALTER TABLE DegreeEarned ADD CONSTRAINT DEGREE_DEGREEEARNED_FK
FOREIGN KEY (DegreeID)
REFERENCES Degree (DegreeID)
NOT DEFERRABLE;

ALTER TABLE invitation ADD CONSTRAINT EVENT_INVITATION_FK
FOREIGN KEY (eID)
REFERENCES Event (eID)
NOT DEFERRABLE;

ALTER TABLE Mentor ADD CONSTRAINT CSUSTUDENTS_MENTOR_FK
FOREIGN KEY (CSUStudentID)
REFERENCES CSUStudents (CSUStudentID)
NOT DEFERRABLE;

ALTER TABLE AlumniEmployment ADD CONSTRAINT COMPANY_ALUMNIEMPLOYMENT_FK
FOREIGN KEY (CompanyID)
REFERENCES Company (CompanyID)
NOT DEFERRABLE;

ALTER TABLE Donations ADD CONSTRAINT FUNDS_DONATIONS_FK
FOREIGN KEY (FundID)
REFERENCES Funds (FundID)
NOT DEFERRABLE;

ALTER TABLE aExpertise ADD CONSTRAINT EXPERTISE_AEXPERTISE_FK
FOREIGN KEY (ExID)
REFERENCES Expertise (ExID)
NOT DEFERRABLE;

ALTER TABLE Mentor ADD CONSTRAINT ALUMNI_MENTOR_FK
FOREIGN KEY (AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;
```

```
ALTER TABLE aExpertise ADD CONSTRAINT ALUMNI_AEXPERTISE_FK
FOREIGN KEY (AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;

ALTER TABLE AlumniEmployment ADD CONSTRAINT ALUMNI_ALUMNIEMPLOYMENT_FK1
FOREIGN KEY (Alumni_AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;

ALTER TABLE Donations ADD CONSTRAINT ALUMNI_DONATIONS_FK
FOREIGN KEY (AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;

ALTER TABLE DegreeEarned ADD CONSTRAINT ALUMNI_DEGREEEARNED_FK
FOREIGN KEY (AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;

ALTER TABLE invitation ADD CONSTRAINT ALUMNI_INVITATION_FK
FOREIGN KEY (AlumniID)
REFERENCES Alumni (AlumniID)
NOT DEFERRABLE;
```

LOGICAL DESIGN SCHEMA

For the established entities, we concluded the attributes and the keys to be used (primary key is underlined, foreign key in bold)

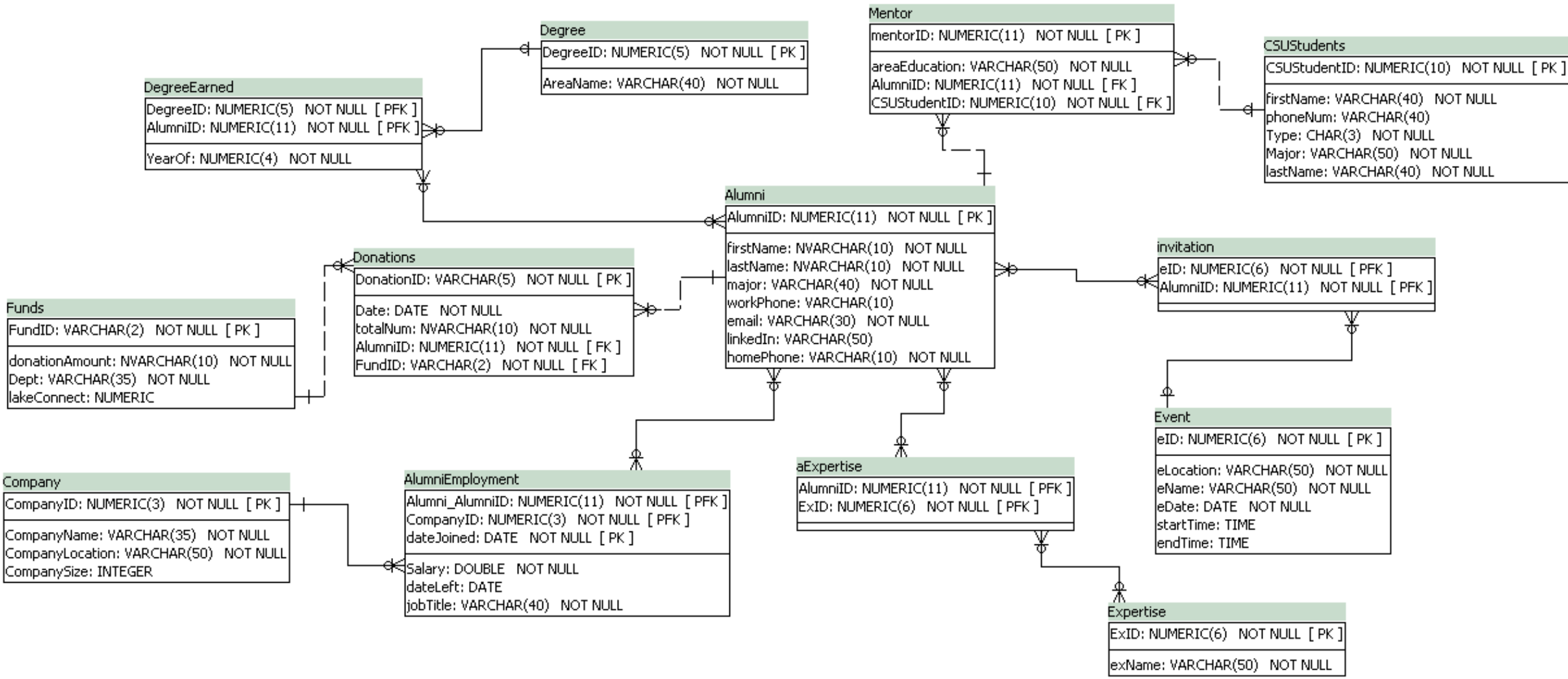
Alumni	Alumni(AlumniID, fname, lname, major, wphone, hphone, email, linkedIn)
CSUStudents	CSUStudents(CSUStudentID, fname, phoneNum, Type, Major, lname)
Mentor	Mentor(mentorID, areaEducation, **AlumniID, CSUStudentID**)
Company	Company(CompanyID, CompanyName, CompanyLocation,CompanySize)
AlumniEmployment	AlumniEmployment(AlumniID, CompanyID, dateJoined, salary, dateLeft, jobTitle)
Donations	Donations(DonationID, date, totalNum, **AlumniID, FundID**)
Funds	Funds(FundID, donationAmount, Dept, lakeConnect)

LOGICAL DESIGN SCHEMA

For the determined entities, we concluded the attributes and the keys to be used (primary key is underlined, foreign keys in bold)

Event	Event(<u>eID</u>, eLocation, eName, eDate, startTime, endTime)
Invitation	Invitation(**eID, AlumniID**)
Expertise	Expertise(<u>ExID</u>, exName)
Aexpertise	AExpertise(**AlumniID, ExI**D)
Degree	Degree(<u>DegreeID</u>, areaName)
DegreeEarned	DegreeEarned(**DegreeId, AlumniID**, yearOf)

DATABASE STRUCTURE



IMPLEMENTATION

Of Database

CREATING THE DATABASE

After completing the physical aspect of our model, we can use the schema provided by the SQL Power Architect.

The script creates all the relations and implements all constraints into Oracle.

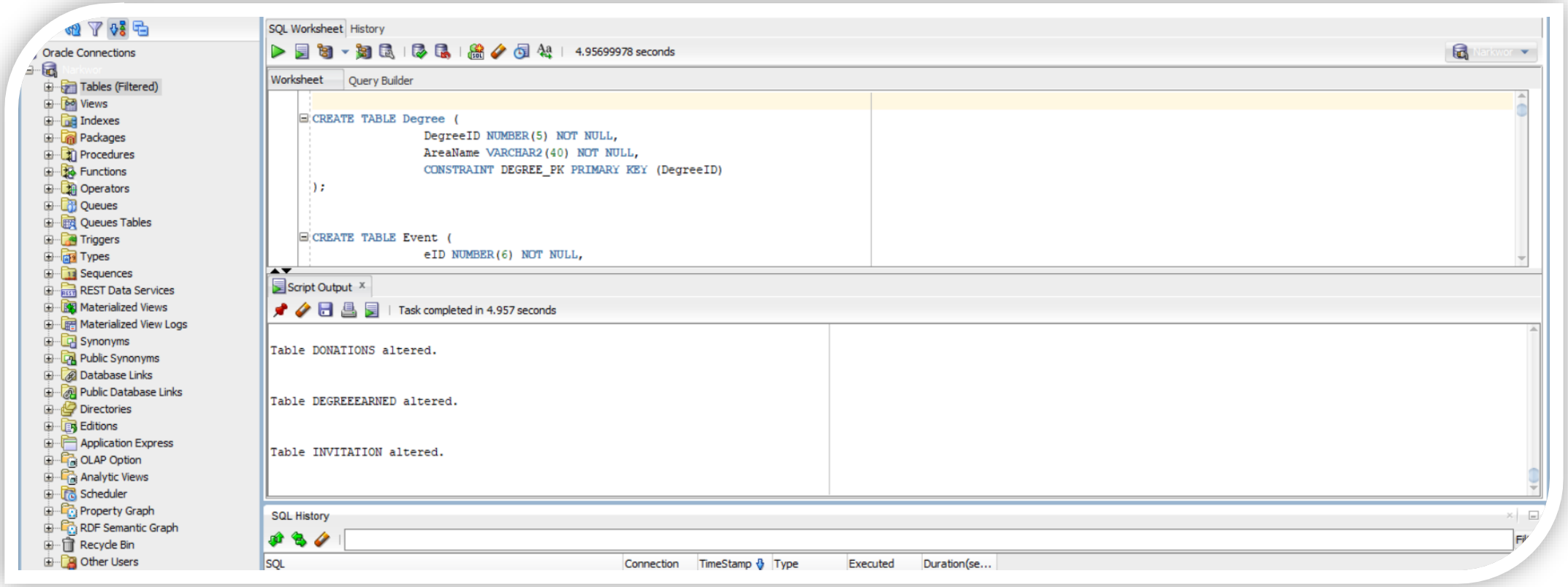
With the database completely implemented, we can add the data to perform queries to create new relations.

DATABASE CREATION

```
CREATE TABLE Degree (
                DegreeID NUMBER(5) NOT NULL,
                AreaName VARCHAR2(40) NOT NULL,
                CONSTRAINT DEGREE_PK PRIMARY KEY (DegreeID)
);

CREATE TABLE Event (
                eID NUMBER(6) NOT NULL,
                eLocation VARCHAR2(50) NOT NULL,
                eName VARCHAR2(50) NOT NULL,
                eDate DATE NOT NULL,
                startTime DATE,
                endTime DATE,
                CONSTRAINT EVENT_PK PRIMARY KEY (eID)
);

CREATE TABLE CSUStudents (
                CSUStudentID NUMBER(10) NOT NULL,
                firstName VARCHAR2(40) NOT NULL,
                phoneNum VARCHAR2(40),
                Type CHAR(3) NOT NULL,
                Major VARCHAR2(50) NOT NULL,
                lastName VARCHAR2(40) NOT NULL,
                CONSTRAINT CSUSTUDENTS_PK PRIMARY KEY (CSUStudentID)
);

CREATE TABLE Company (
                CompanyID NUMBER(3) NOT NULL,
                CompanyName VARCHAR2(35) NOT NULL,
                CompanyLocation VARCHAR2(50) NOT NULL,
                CompanySize NUMBER DEFAULT NULL,
                CONSTRAINT COMPANY_PK PRIMARY KEY (CompanyID)
);
COMMENT ON COLUMN Company.CompanyName IS 'company name';
COMMENT ON COLUMN Company.CompanyLocation IS 'company  address';
COMMENT ON COLUMN Company.CompanySize IS 'company employee size';

CREATE TABLE Funds (
                FundID VARCHAR2(2) NOT NULL,
                donationAmount NVARCHAR2(10) DEFAULT 'amount' NOT NULL,
                Dept VARCHAR2(35) DEFAULT 'Department name' NOT NULL,
                lakeConnect NUMBER,
                CONSTRAINT FUNDS_PK PRIMARY KEY (FundID)
);
COMMENT ON COLUMN Funds.lakeConnect IS 'Yes or No accepting laker events

CREATE TABLE Expertise (
                ExID NUMBER(6) NOT NULL,
                exName VARCHAR2(50) NOT NULL,
                CONSTRAINT EXPERTISE_PK PRIMARY KEY (ExID)
);

CREATE TABLE Alumni (
                AlumniID NUMBER(11) NOT NULL,
                firstName NVARCHAR2(10) NOT NULL,
                lastName NVARCHAR2(10) NOT NULL,
                major VARCHAR2(40) NOT NULL,
                workPhone VARCHAR2(10) DEFAULT NULL,
                email VARCHAR2(30) NOT NULL,
                linkedIn VARCHAR2(50) DEFAULT NULL,
                homePhone VARCHAR2(10) NOT NULL,
                CONSTRAINT ALUMNI_PK PRIMARY KEY (AlumniID)
);

CREATE TABLE invitation (
                eID NUMBER(6) NOT NULL,
                AlumniID NUMBER(11) NOT NULL,
                CONSTRAINT INVITATION_PK PRIMARY KEY (eID, AlumniID)
);

CREATE TABLE Donations (
                DonationID VARCHAR2(5) NOT NULL,
                Date_1 DATE NOT NULL,
                totalNum NVARCHAR2(10) NOT NULL,
                AlumniID NUMBER(11) NOT NULL,
                FundID VARCHAR2(2) NOT NULL,
                CONSTRAINT DONATIONS_PK PRIMARY KEY (DonationID)
);

CREATE TABLE AlumniEmployment (
                Alumni_AlumniID NUMBER(11) NOT NULL,
                CompanyID NUMBER(3) NOT NULL,
                dateJoined DATE NOT NULL,
                Salary NUMBER NOT NULL,
                dateLeft DATE DEFAULT NULL,
                jobTitle VARCHAR2(40) NOT NULL,
                CONSTRAINT ALUMNIEMPLOYMENT_PK PRIMARY KEY (Alumni_AlumniID, CompanyID, dateJoined)
);
COMMENT ON COLUMN AlumniEmployment.Alumni_AlumniID IS 'Alumni identification';
COMMENT ON COLUMN AlumniEmployment.dateJoined IS 'Date Alumni worked within company';

CREATE TABLE aExpertise (
                AlumniID NUMBER(11) NOT NULL,
                ExID NUMBER(6) NOT NULL,
                CONSTRAINT AEXPERTISE_PK PRIMARY KEY (AlumniID, ExID)
);
```

```
CREATE TABLE Mentor (
                mentorID NUMBER(11) NOT NULL,
                education VARCHAR2(50) NOT NULL,
                AlumniID NUMBER(11) NOT NULL,
                CSUStudentID NUMBER(10) NOT NULL,
                CONSTRAINT MENTOR_PK PRIMARY KEY (mentorID)
);

CREATE TABLE DegreeEarned (
                DegreeID NUMBER(5) NOT NULL,
                AlumniID NUMBER(11) NOT NULL,
                YearOf NUMBER(4) NOT NULL,
                CONSTRAINT DEGREEEARNED_PK PRIMARY KEY (DegreeID, AlumniID)
);
```

IMPLEMENTATION DATABASE AND INSERTION



[Alumni_DB_Screenshot.pptx]

SQL QUESTION 1

Request

A listing of all donors who have donated with name, graduation date, major, and a total amount. The report will be sorted in descending order of the total donated amount

Query

```
SELECT firstname, lastname,
major, yearof, SUM(totalnum) AS
"Total Donated Amount"
FROM Alumni
JOIN DegreeEarned
USING(AlumniID)
JOIN Donations USING(AlumniID)
GROUP BY(firstname, lastname,
major, yearof)
ORDER BY("Total Donated
Amount") DESC;
```

Relation

	FIRSTNAME	LASTNAME	MAJOR	YEAROF	Total Donated Amount
1	Jane	Foster	computer science	2019	400
2	John	Doe	computer science	2016	300

SQL QUESTION 2

Request

A list of the specified area and year of graduation, each alumnus' name, e-mail address, the degree earned (eg. BS, MBA, etc.), work phone number, home phone number, and LinkedIn, who graduated between 2000 and 2020.

Query

```
SELECT yearOf, firstname,
lastname, email, areaname,
workphone, homephone, linkedin
FROM Alumni
JOIN DegreeEarned
USING(AlumniID)
JOIN Degree USING(DegreeID)
WHERE yearOf BETWEEN 2000 AND
2020;
```

Relation

	YEAROF	FIRSTNAME	LASTNAME	EMAIL	AREANAME	WORKPHONE	HOMEPHONE	LINKEDIN
1	2019	Jane	Foster	janefoster2@gmail.com	Computer Science, BS	(null)	4042561031	(null)
2	2016	John	Doe	johndoe1@yahoo.com	Computer Science, BS	(null)	4040001001	(null)

SQL QUESTION 3

Request

A list of alumni who work in Norfolk Southern with name, join date, title, and salary range.

Query

```
SELECT firstname, lastname, datejoined, jobtitle, salary FROM Alumni A
JOIN AlumniEmployment AE
ON(A.alumniID = AE.alumni_alumniID)
JOIN Company USING(CompanyID)
WHERE CompanyName = 'Norfolk Southern';
```

Relation

	FIRSTNAME	LASTNAME	DATEJOINED	JOBTITLE	SALARY
1	Jonathan	Arnold	11-AUG-20	Marketing Manager	65000

SQL QUESTION 4

Request

Find all alumni whose expertise is in 'Database' and 'Software development'.

Query

```
SELECT Alumni.*
FROM Alumni WHERE alumniID IN(
SELECT alumniID FROM AExpertise
JOIN Expertise USING(exID)
WHERE exName = 'Database' )
AND alumniID IN( SELECT
alumniID
FROM AExpertise
JOIN Expertise USING(exID)
WHERE exName = 'Software
Development');
```

Relation

	ALUMNIID	FIRSTNAME	LASTNAME	MAJOR	WORKPHONE	EMAIL	LINKEDIN	HOMEPHONE
1	2	Jane	Foster	Computer Science	(null)	janefoster2@gmail.com	(null)	4042561031

SQL QUESTION 5

Request

Find all alumni who mentor current CSU student, list alumni's name, student's name.

Query

```
SELECT a.firstname AS "Alumni FirstName", a.lastname AS "Alumni LastName", csu.firstname AS "Student FirstName", csu.lastname AS "Student LastName"
FROM Alumni a, Mentor, CSUStudents csu
WHERE a.alumniID = mentor.alumniID AND mentor.csustudentID = csu.csustudentID;
```

Relation

	Alumni FirstName	Alumni LastName	Student FirstName	Student LastName
1	Jane	Foster	Edward	Elrick
2	Jane	Foster	Patrick	Starfish

SUMMARY

CSU Alumni is a database that allows the public department to write a digital profile for Clayton State University alumni. With the Alumni profile being digitized, the public department/university can now to keep experiences of CSU Alumni employment statuses. The database report includes degree earned, year of graduation, emails, work phone numbers of alumni, mentors and those who have interest in mentorship with CSU Alumni. Voluntary email of CSU Alumni will increase engagement by linking CSU Alumni to events and programs happening at Clayton State University. In addition to emails, donations to Clayton State University have reference numbers, so these donations can be tracked and reported on.

Overall CSU Alumni database allows Clayton State University to maintain the records of CSU Alumni and provide both students and alumni for an easy-to-use mentorship experience.

THANK YOU

Alphonso Anderson

Celso Fuentes

Dennis Thomas

Narkwor Mensah