**Database Project Final Report**

**CSE 3241 – Introduction to**

**Database Systems**

**Leon Madrid**

**Tuesday/Thursday, 9:35-10:55 am**

Tyler Cingel, Lily Driscoll, Isaac Mattern, Jacob Woodhouse

Section 1: Database Description

**Figure 1: Entity-Relation Diagram**

Diagram

Description automatically generated

**Figure 2: Relational Schema**

Timeline

Description automatically generated

**Normalization achieved for each table:**

Media Item –

Media Genres – Because this table only has three attributes which make its key, and there is no dependency between any of them, it is in BCNF

Artist Genres – This is in BCNF because, while there may be a trend, there is no dependence between Name and Genres, and both of these attributes create the primary key

Branch – Because this table only has two attributes which make its key and there is no dependency between city and state, it is in BCNF

Order – This is in BCNF because non-key attributes are dependent on the key and the key alone

Patron – This is in BCNF because non-key attributes are dependent on the key and the key alone

Checkouts – This is in BCNF because non-key attributes are dependent on the key and the key alone

Chapter – This is in BCNF because non-key attributes are dependent on all key attributes and nothing else

Author – This table only has two attributes, one of which is the key, and the other of which is dependent on the key, meaning this table is in BCNF

Actor – This is in BCNF because non-key attributes are dependent on the key and the key alone

Game Studio – Because this table only has one attribute (Name) it is in BCNF

Artist – This is in BCNF because non-key attributes are dependent on the key and the key alone

Track – This is in BCNF because non-key attributes are dependent on both key attributes and nothing else

Author Writes – This is in BCNF because non-key attributes are dependent on both key attributes and nothing else

Actor Stars – This is in BCNF because non-key attributes are dependent on both key attributes and nothing else

Studio Creates – This is in BCNF because non-key attributes are dependent on both key attributes and nothing else

Artist Authors – This is in BCNF because non-key attributes are dependent on both key attributes and nothing else

Album Contains – This is in BCNF because all attributes in the table create the key and no dependencies exist between them

**Indexes included:**

1. CREATE INDEX track\_artist

ON Track (Artist\_Name);

There will likely be many instances where we want to retrieve tracks only by a specific artist.

1. CREATE INDEX patron\_checkouts

ON Checkouts (Email\_Address);

It would be useful for the checkouts of specific patrons to be indexed so that when individual (or all) checkouts of a specific patron are needed, they are quickly accessible.

1. CREATE INDEX best\_available\_media

ON Media\_Item (Rating, Status, City);

Many patrons will only care to see the media items which are available and in a specific city. Additionally, many patrons would like to only see media items above a certain rating.

**Views included:**

1. Total number of chapters in each book to help users gauge the size of the book
   1. CREATE VIEW BookChaptersTotal As

SELECT Media\_Item.Name, Author\_Writes.Name as Author, COUNT (DISTINCT Chapter.Name) as Total\_Chapters

FROM Media\_Item, Author\_Writes, Chapter

WHERE Author\_Writes.ID=Chapter.ID AND Audiobook\_Flag=1 AND Chapter.ID=Media\_Item.ID

GROUP By Media\_Item.Name

* 1. π media\_item.name, author\_writes.name → author, COUNT (\delta name) → total\_chapters

γ name, COUNT (chapter.name)

σ author\_writes.id = chapter.id AND audiobook\_flag = 1 AND chapter.id = media\_item.id (media\_item × author\_writes × chapter)

* 1. Table

     Description automatically generated

1. Total number of checkouts by patron to help users see the most active patrons
   1. CREATE VIEW CheckoutsMade AS

SELECT First\_Name, Last\_Name, COUNT(Checkout\_ID) as Total\_Checkedout

FROM Patron, Checkouts

Where Patron.Email\_Address=Checkouts.Email\_Address

GROUP BY Last\_Name;

* 1. δ π first\_name, last\_name, COUNT (checkout\_id) → total\_checkedout

ρ last\_name, COUNT (checkout\_id)

σ patron.email\_address = checkouts.email\_address (patron × checkouts)

* 1. Table

     Description automatically generated

**Transaction samples:**

1. **Move library stock to new branch**

BEGIN TRANSACTION Move\_Stock

UPDATE Media\_Item SET City=new\_city, State=new\_state WHERE City=old\_city, State=old\_state;

IF error THEN GO TO UNDO; END IF;

COMMIT;

GO TO FINISH;

UNDO:

ROLLBACK;

FINISH;

END TRANSACTION;

1. **A customer signs up to become a patron**

BEGIN TRANSACTION New\_Patron

INSERT INTO Patron VALUES (email, fname, lname, address, city, state);

IF error THEN GO TO UNDO; END IF;

COMMIT;

GO TO FINISH;

UNDO:

ROLLBACK;

FINISH;

END TRANSACTION;

1. **A patron checks out a media item**

BEGIN TRANSACTION Patron\_Checkout

INSERT INTO Checkout VALUES (Get\_ChID(), DATE.Current + 7\*DATE.Day, DATE.Current, Patron.Email\_Address, Book.ID, Book.Copy#)

IF error THEN GO TO UNDO; END IF;

COMMIT;

GO TO FINISH;

UNDO:

ROLLBACK;

FINISH;

END TRANSACTION;

Section 2: User Manual

**Table descriptions:**

Media Item – A single piece of any media (audiobook, album, movie, game) in the library system

ID – Numeric ID of the media item, CHAR(10), NOT NULL, Primary Key

Copy# - Which copy of media item out of all copies, INT, NOT NULL, Primary Key

Name – Name of media item, VARCHAR (50)

Year – Year of release of item, INT

Length – Length of media item (minutes), INT

Audiobook\_Flag – Flag for type of media (1 if item is an audiobook, 0 otherwise), INT

Album\_Flag – Flag for type of media (1 if item is an album, 0 otherwise), INT

Movie\_Flag – Flag for type of media (1 if item is a movie, 0 otherwise), INT

Game\_Flag – Flag for type of media (1 if item is a game, 0 otherwise), INT

Publisher – Publishing company of media if applicable, VARCHAR(15)

Director – Director of media if applicable, VARCHAR(20)

Console – Console media is played on if applicable, VARCHAR(20)

Rating – Rating of media item, VARCHAR(5)

Status – Flag for media’s availability (1 if item is available, 0 otherwise), INT

Order# - If currently being shipped, number for order, INT, Foreign Key (Orders)

City – City the branch owning the item is in, VARCHAR(20), NOT NULL, Foreign Key (Branch)

State – State the branch owning the item is in, VARCHAR(20), NOT NULL, Foreign Key (Branch)

Media Genres – List of genres for any piece of media (there can be multiple per item)

ID – Numeric ID of the media item, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Genres – One genre for this specific media item, VARCHAR(20), NOT NULL, Primary Key

Artist Genres – List of genres an artist makes music for (there can be multiple genres per artist)

Name – Name of artist, VARCHAR(50), NOT NULL, Foreign Key (Artist), Primary Key

Genres – Specific Genre, VARCHAR(20), NOT NULL, Primary Key

Branch – A branch of this library represented by the city and state it’s in

City – City where library is located, VARCHAR(20), NOT NULL, Primary Key

State – State where library is located, VARCHAR(20), NOT NULL, Primary Key

Orders – An order that is in-transit to a given library

Order# - Number for the order, INT, NOT NULL, Primary Key

Price – Total cost of the order, INT

Quantity – Number of items ordered, INT

Arrival Date – Date when items will arrive, DATE

City – City where destination library is located, VARCHAR(20), NOT NULL, Foreign Key (Branch)

State – State where destination library is located, VARCHAR(20), NOT NULL Foreign Key (Branch)

Patron – A customer of the library that can check out media

Email Address – Email address associated with individual, VARCHAR(50), NOT NULL, Primary Key

First Name – First name of individual, VARCHAR(20), NOT NULL

Last Name – Last name of individual, VARCHAR(20), NOT NULL

Address – Home address of individual, VARCHAR(50)

City – City where patron’s branch is located, VARCHAR(20), NOT NULL, Foreign Key (Branch)

State – State where patron’s branch is located, VARCHAR(20), NOT NULL, Foreign Key (Branch)

Checkouts – A log of a checked out item

CheckoutID – Unique ID associated with checkout log, VARCHAR(50), NOT NULL, Primary Key

Due Date – Date the item is/was due, DATE, NOT NULL

Checkout Date – Date the item was checked out, DATE, NOT NULL

Email Address – Patron’s email address, VARCHAR(50), NOT NULL, Foreign Key (Patron)

ID – ID of the checked out media item, CHAR(10), NOT NULL, Foreign Key (Media\_Item)

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item)

Chapter – A single chapter of an audiobook in the library

Name – Name of chapter, VARCHAR(50), NOT NULL, Primary Key

Number – Number of the chapter in the book in which it appears, INT, NOT NULL, Primary Key

ID – ID of the media item containing the chapter, CHAR(10), NOT NULL, Foreign Key (Media\_Item)

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item)

Author – An author in the database

Name – Name of author, VARCHAR(50), NOT NULL, Primary Key

Age – Age of author, INT

Actor – An actor in the database

Name – Name of actor, VARCHAR(50), NOT NULL, Primary Key

Sex – Gender of actor (‘M’, ‘F’), CHAR(1)

Age – Age of actor, INT

Game Studio – A studio that specializes in game development in the database

Name – Name of game studio, VARCHAR(50), NOT NULL, Primary Key

Artist – An artist in the database

Name – Name of artist, VARCHAR(50), NOT NULL, Primary Key

Sex – Gender of artist (‘M’, ‘F’), CHAR(1)

Age – Age of artist, INT

Track – A single song in the database

Name – Name of song, VARCHAR(50), NOT NULL, Primary Key

Genre – Genre of the song, VARCHAR(20)

Artist Name – Name of artist who made song, VARCHAR(50), NOT NULL, Foreign Key (Artist)

Author Writes – Link between a book and the author that wrote it

Name – Name of author, VARCHAR(50), NOT NULL, Primary Key

ID – ID of the book, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Actor Stars – Link between an actor and the movie they’re starring in

Name – Name of actor, VARCHAR(50), NOT NULL, Primary Key

ID – ID of the movie, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Studio Creates – Link between a studio and the game they’re creating

Name – Name of studio, VARCHAR(50), NOT NULL, Primary Key

ID – ID of the game, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Artist Authors – Link between an artist and the album they’re creating

Name – Name of artist, VARCHAR(50) , NOT NULL, Primary Key

ID – ID of the album, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Album Contains – List of tracks within an album

ID – ID of the song, CHAR(10), NOT NULL, Foreign Key (Media\_Item), Primary Key

Copy# - Which copy of the media item, INT, NOT NULL, Foreign Key (Media\_Item), Primary Key

Name – Name of track in album, VARCHAR(50), NOT NULL, Foreign Key (Track), Primary Key

Artist Name – Name of artist who made album, VARCHAR(50), NOT NULL, Foreign Key (Track), Primary Key

**Sample SQL Queries:**

Find the titles of all tracks by ARTIST released before YEAR

* SELECT DISTINCT Name FROM Album\_Contains, Media\_Item WHERE Media\_Item.Year<  
  YEAR AND Album\_Contains.ArtistName=ARTIST;
* π[Name](σ[Year < YEAR](Media\_Item) X σ[ArtistName = ARTIST](Album\_Contains))

Give all the movies and their date of their checkout from a single PATRON

* SELECT Name, Checkout\_Date FROM Media\_Item, Checkouts WHERE Email\_address=PATRON AND Movie\_Flag=1 AND Media\_Item.ID = Checkouts.ID AND Media\_Item.Copy\_Number = Checkouts.Copy\_Number;
* π[Name, Checkout\_Date](σ[Movie\_Flag = 1](Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Email\_Address = PATRON](Checkouts))

List all the albums and their unique identifiers with less than 2 copies held by the library

* SELECT Name, ID FROM Media\_Item WHERE Album\_Flag=1 GROUP BY ID HAVING COUNT(\*) < 2;
* π[Name, ID](σ[Album\_Flag = 1] AND [F\_COUNT(ID) < 2](Media\_Item))

Give all the patrons who checkout out a movie by ACTOR and the movies they checked out

* SELECT First\_Name, Last\_Name, Media\_Item.Name FROM Patron, Media\_Item, Actor\_Stars, Checkouts WHERE Actor\_Stars.Name = ACTOR AND Actor\_Stars.ID = Media\_Item.ID AND Actor\_Stars.Copy\_Number = Media\_Item.Copy\_Number AND Checkouts.Email\_Address = Patron.Email\_Address AND Media\_Item.ID = Checkouts.ID AND Media\_Item.Copy\_Number = Checkouts.Copy\_Number;
* π[First\_Name, Last\_Name, Name]((Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Name = ACTOR](Actor\_Stars) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number](Checkouts) ⋈ [Email\_Address = Email\_Address](Patron))

Find the total number of albums checked out by a single PATRON

* SELECT COUNT(\*) FROM Media\_Item, Checkouts WHERE Checkouts.Email\_Address=PATRON AND Album\_Flag=1 AND Media\_Item.ID = Checkouts.ID AND Media\_Item.Copy\_Number = Checkouts.Copy\_Number;
* F\_COUNT(σ[Album\_Flag = 1](Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Email\_Address = PATRON](Checkouts))

Find the patron who has checked out the most videos and the total number of videos they have checked out

* SELECT Patron.First\_Name, Patron.Last\_Name, COUNT(\*) FROM Patron, Media\_Item, Checkouts WHERE Checkouts.Email\_Address = Patron.Email\_Address AND Movie\_Flag=1 AND Media\_Item.ID = Checkouts.ID AND Media\_Item.Copy\_Number = Checkouts.Copy\_Number GROUP BY Checkouts.Email\_Address ORDER BY COUNT(\*) DESC LIMIT 1;
* π[First\_Name, Last\_Name, F\_COUNT(ID)](σ[Movie\_Flag = 1](Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Email\_Address = PATRON](Checkouts) ⋈[Email\_Address =   
  Email\_Address] (Patron))

List all the games from a game STUDIO

* SELECT DISTINCT Media\_Item.Name FROM Media\_Item, Studio\_Creates WHERE Game\_Flag=1 AND Studio\_Creates.Name=STUDIO AND Media\_Item.ID = Studio\_Creates.ID AND Media\_Item.Copy\_Number = Studio\_Creates.Copy\_Number;
* π[Name](σ[Game\_Flag = 1](Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Name = STUDIO](Studio\_Creates))

Find how many movies an ACTOR has starred in that are RATING and are in stock

* SELECT COUNT(\*) FROM Media\_Item, Actor\_Stars WHERE Rating=RATING AND Movie\_Flag=1 AND Status = 1 AND Actor\_Stars.Name = ACTOR AND Actor\_Stars.ID = Media\_Item.ID AND Media\_Item.Copy\_Number = Actor\_Stars.Copy\_Number;

F\_COUNT(σ [Rating = RATING] AND [Movie\_Flag = 1] AND [Status = 1](Media\_Item) ⋈[ID = ID] AND [Copy\_Number = Copy\_Number] σ[Name = ACTOR](Actor\_Stars))**INSERT Statement descriptions/examples**

Albums

Syntax – INSERT INTO Media\_Item VALUES ([ID], [copy#], [name], [year], [length], 0, 1, [publisher], 0, NULL, 0, NULL, [rating], [status], [order#], [city], [state]);

Dependencies – Branch

Example – INSERT INTO Media\_Item VALUES (‘1265472654’, 1, ‘Greatest Hits’, 1985, 113, 0, 1, ‘Columbia’, 0, NULL, 0, NULL, NULL, 1, NULL, ‘Cleveland’, ‘Ohio’);

Movies/Videos

Syntax – INSERT INTO Media\_Item VALUES ([ID], [copy#], [name], [year], [length], 0, 0, NULL, 1, [director], 0, NULL, [rating], [status], [order#], [city], [state]);

Dependencies – Branch

Example – INSERT INTO Media\_Item VALUES (‘1358594289’, 3, ‘Star Wars IV: A New Hope’, 1977, 121, 0, 0, NULL, 1, ‘George Lucas’, 0, NULL, NULL, 1, NULL, ‘Phoenix’, ‘Arizona’);

Audiobooks

Syntax – INSERT INTO Media\_Item VALUES ([ID], [copy#], [name], [year], [length], 1, 0, [publisher], 0, NULL, 0, NULL, [rating], [status], [order#], [city], [state]);

Dependencies – Branch

Example – INSERT INTO Media\_Item VALUES (‘1234567890’, 1, ‘Harry Potter and the Sorcerer’s Stone’, 1997, 269, 0, 0, ‘Bloomsbury’, 0, NULL, 0, NULL, NULL, 1, NULL, ‘Columbus’, ‘Ohio’);

Artists

Syntax – INSERT INTO Artist VALUES ([name], [gender(F/M)], [age]);

Dependencies – N/A

Example – INSERT INTO Artist VALUES (‘Bruno Mars’, ‘M’, ‘36’);

Patrons

Syntax – INSERT INTO Patron VALUES ([email], [first name], [last name], [address], [city], [state]);

Dependencies – N/A

Example – INSERT INTO Patron VALUES (‘email@yahoo.com’, ‘John’, ‘Doe’, ‘404 Missing Lane, Las Vegas, NV 44720’, ‘Las Vegas’, ‘Nevada’);

**Miscellaneous INSERT Examples**

Branch – INSERT INTO Branch VALUES ('Cleveland', 'Ohio');

Orders – INSERT INTO Orders VALUES (2011, 1500, 1, 2022-01-04, 'Columbus', 'Ohio');

Checkouts – INSERT INTO Checkouts VALUES ('1111111111', '2021-11-08', '2021-10-08', 'librarypatron@gmail.com', '6421358790', 1);

Author – INSERT INTO Author VALUES ('Joanne K. Rowling', 56);

Actor – INSERT INTO Actor VALUES ('Mark Hamill', 'M', 70);

Game Studio – INSERT INTO Game\_Studio VALUES ('NineHertz');

Artist Genre – INSERT INTO Artist\_Genres VALUES ('Coldplay', 'Rock');

Track – INSERT INTO Track VALUES ('Politik', 'Rock', 'Coldplay');

Media Genres – INSERT INTO Media\_Genres VALUES ('1234567890',1, 'Fantasy');

Chapter – INSERT INTO Chapter VALUES ('The Boy Who Lived', 1,'1234567890',1);

Author Writes – INSERT INTO Author\_Writes VALUES ('Patrick James Rothfuss', '4563153215',1);

Actor Stars – INSERT INTO Actor\_Stars VALUES ('Mark Hamill', '7678906543',1);

Studio Creates – INSERT INTO Studio\_Creates VALUES ('Nintendo', '6421358790',1);

Artist\_Authors – INSERT INTO Artist\_Authors VALUES ('Coldplay', '2469762156',1);

Album Contains – INSERT INTO Album\_Contains VALUES ('Politik', '2469762156', 1, 'Coldplay');

**DELETE Statement descriptions/examples**

Albums

Syntax – DELETE FROM Media\_Item WHERE Album\_Flag=1 AND ID=?;

Dependencies – Album\_Contains, Artist\_Authors

Example – DELETE FROM Media\_Item WHERE Album\_Flag=1 AND ID = ‘1265472654’;

Movies/Videos

Syntax – DELETE FROM Media\_Item WHERE MovieFlag=1 AND ID=?;

Dependencies – Actor\_Stars

Example – DELETE FROM Media\_Item WHERE MovieFlag=1 AND ID=’7678906543’;

Audiobooks

Syntax – DELETE FROM Media\_Item WHERE AudiobookFlag=1 AND ID=?;

Dependencies – Author\_Writes

Example – DELETE FROM Media\_Item WHERE AudiobookFlag=1 AND ID=’1234567890’;

Artists

Syntax – DELETE FROM Artists WHERE Name=?;

Dependencies – Track, Artist\_Authors, Artist\_Genres

Example – DELETE FROM Artists WHERE Name=’Bruno Mars’;

Patrons

Syntax – DELETE FROM Patron WHERE Email\_Address=?;

Dependencies – Checkouts

Example – DELETE FROM Patron WHERE Email\_Address=’libraryuser@yahoo.com’;

**Miscellaneous DELETE Examples**

Branch – DELETE FROM Branch WHERE City=’Houston’ AND State=’Texas’;

Orders – DELETE FROM Orders WHERE Arrival\_Date<DATE.current;

Checkouts – DELETE FROM Checkouts WHERE Checkout\_Date+[7 days]<DATE.current;

Author – DELETE FROM Author WHERE Name=’Dr. Seuss’;

Actor – DELETE FROM Actor WHERE Name=’Carrie Fisher’;

Game Studio – DELETE FROM Game\_Studio WHERE Name=’Electronic Arts’;

Artist Genre – DELETE FROM Artist\_Genre WHERE Genres=’Electronic Rock’;

Track – DELETE FROM Track WHERE Name=’Greatest Hits’;

Media Genres – DELETE FROM Media\_Genres WHERE ID=’1234567890’;

Chapter – DELETE FROM Chapter WHERE Name=’Chapter 1’ AND Number=1;

Author Writes – DELETE FROM Author\_Writes WHERE ID=’1234567890’;

Actor Stars – DELETE FROM Actor\_Stars WHERE Name=’Carrie Fisher’;

Studio Creates – DELETE FROM Studio\_Creates WHERE ID=’1234567809’ AND Copy#=’4’;

Artist\_Authors – DELETE FROM Artist\_Authors WHERE ID= ‘1265472654’ AND Copy#=’1’;

Album Contains – DELETE FROM Album\_Contains WHERE Name=’Holiday’ AND ArtistName=’Green Day’;