Entities

University:
The university entity represents each university. It has an associated InstitutionId which serves as the primary key, and has attributes for the university's name, city, and country.

Author:
The author entity represents a researcher. It stores a author_id as the primary key, as well as the author_name, university_name of the university where they did their PhD.

work:
The work table describes researcher's outcomes, such papers, posters, etc. Work table takes the work_id as the primary key to uniquely identify each outcome. Work table also contains title and publication_year of the outcome as the attributes.

Topics:
Topics represent possible topics of research outputs. There is a TopicId which serves as the primary key, and the topic_name and category are the attributes of topics.

Music:
Music table represents the background music which will be played when the user reads the information of the outputs. Music table takes MusicId as the primary key. It also takes Genre and filename as attributes.

Relationships

Author to University: The authors are connected to the university through the university_name and Institution_Name in each table. A person must be associated with one university through their PhD Id. A university can be associated with many people.

Work to Topics: This relationship is many-to-many relationship. Because for each work, it can be involved in multiple topics. For each topic, there should be multiple works in the field as well. Therefore, we create a relational table called topic_work table for it.

Author to work: This relationship is many-to-many relationship. Because for each author, he/she can write multiple works. For each work, there could be multiple authors work on it as well. Therefore, we create a relational table called work_author table for it.

Topics to Music: Each category have exactly one corresponding genre of music. Each music genre can have multiple corresponding categories. Therefore, this is a many-to-one relationship.

Normalization Procedure:

Author_id -> author_id, author_name, university_name
InstitutionId -> InstitutionId, Instituion_Name, City, Country
Work_id -> work_id, title, publication_year
Topic_id -> topic_id, topic_name, category
MusicId ->MusicId, filename, genre

Firstly, we split the FDs to make the right side singleton
Author_id -> author_name
Author_id -> university_name
InstitutionId -> Instituion_Name
InstitutionId -> City
InstitutionId -> Country
Work_id -> title
Work_id -> publication_year
Topic_id -> topic_name
Topic_id -> category
MusicId -> filename
MusicId -> genre

Secondly, we check each FD, we notice that none of them is redundant. Therefore, we should keep all of them.
Thirdly, we combine the FDs which in the format A->B, A->C to A->BC, we have:
Author_id -> author_name, university_name
InstitutionId -> Instituion_Name, City, Country
Work_id -> title, publication_year
Topic_id -> topic_name, category
MusicId -> filename, genre

So we can have the relationships which meet the requirements of 3NF:
A(author_id, author_name, university_name)
B(InstitutionId, Institution_Name, City, Country)
C(work_id, title, publication_year)
D(topic_id, topic_name, category)
E(musicId, filename, genre)
F(author_id, InstitutionId, work_id, topic_id, MusicId)

Relational Schema

Your relational schema should be formatted as follows:
Table-Name (Column1: Domain [PK], Column2:Domain [FK to table.column],
Column3: Domain,...)

PK: Indicates that the column is a primary key for the table
FK: Indicates that the column is a foreign key referencing the primary key of table.column. Domain: INT, Decimal, VARCHAR (X),....

Author(author_id:INT [PK] [FK to work_author.author_id], author_name VARCHAR(50), university_name:VARCHAR(255) [FK to Universities.InstitutionId])

Universities(InstitutionId:INT [PK], Institution_Name:VARCHAR(50), City:VARCHAR(100), Country:VARCHAR(100))

work(work_Id:INT [PK] [FK to work_author.work_Id] [FK to topic_work.work_Id], Title:VARCHAR(100), publication_year:INT)

Work_author(author_id:INT [PK] [FK to Author.author_id], work_id:INT [PK] [FK to work.work_id])

Topics (topic_id:INT [PK] [FK to topic_work.topicId], topic_name:VARCHAR(15), category:VARCHAR(20) [FK to TopicMusic.category])

Topic_work(topicId:INT [PK] [FK to Topics.topic_Id], work_id:INT [PK] [FK to work.work_Id])

TopicMusic(category:VARCHAR(255) [PK] [FK to Topics.category], genre:VARCHAR(255) [FK to Music.genre])

Music(MusicId:INT [PK], Genre:VARCHAR(30) [FK to TopicMusic.genre], filename:VARCHAR(30))