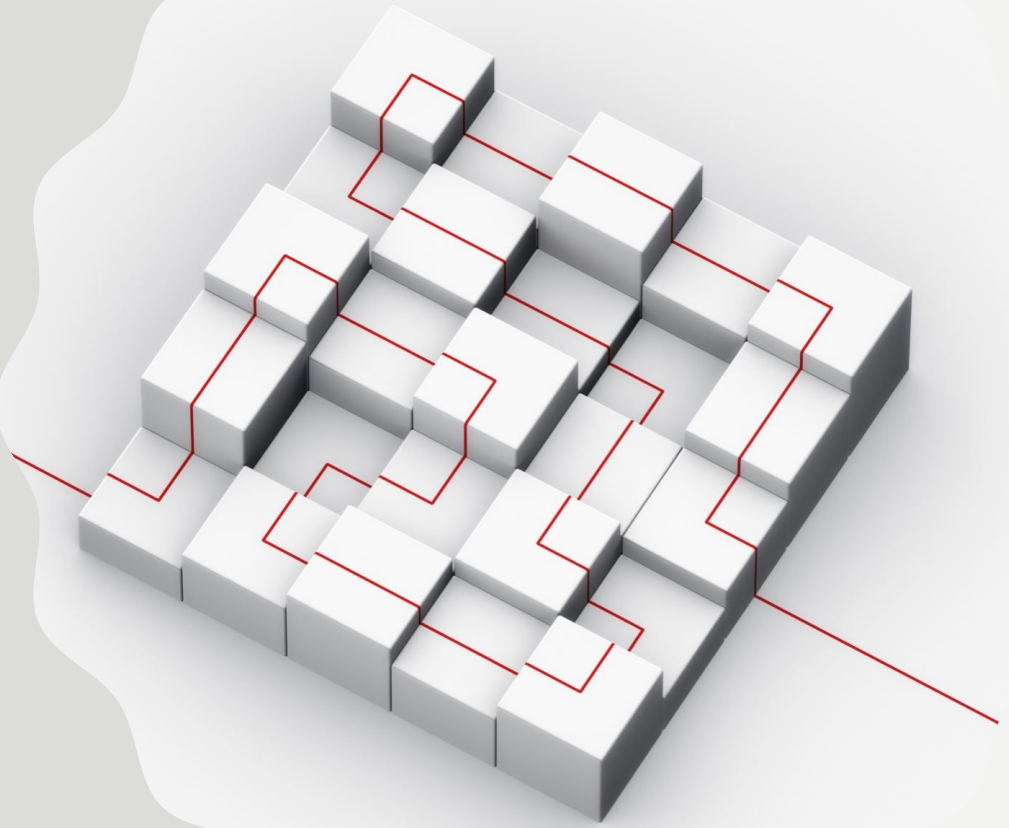
ENTERTAINMENT INDUSTRY MANAGEMENT SYSTEM (EIMS)

Omkar Narkar | Rishabh Madani

CONTENT

- Problem Overview
- Objective
- Data Models
  - EER Diagram
  - UML Diagram
  - SQL Reverse Engineered Diagram
- Analytics

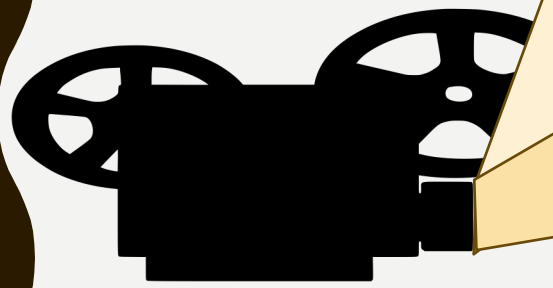


PROBLEM OVERVIEW

The entertainment industry is currently valued at 2.83 Trillion Dollars. However, the industry lacks centralized control due to its expansive and fragmented landscape, struggling to maximize its potential.

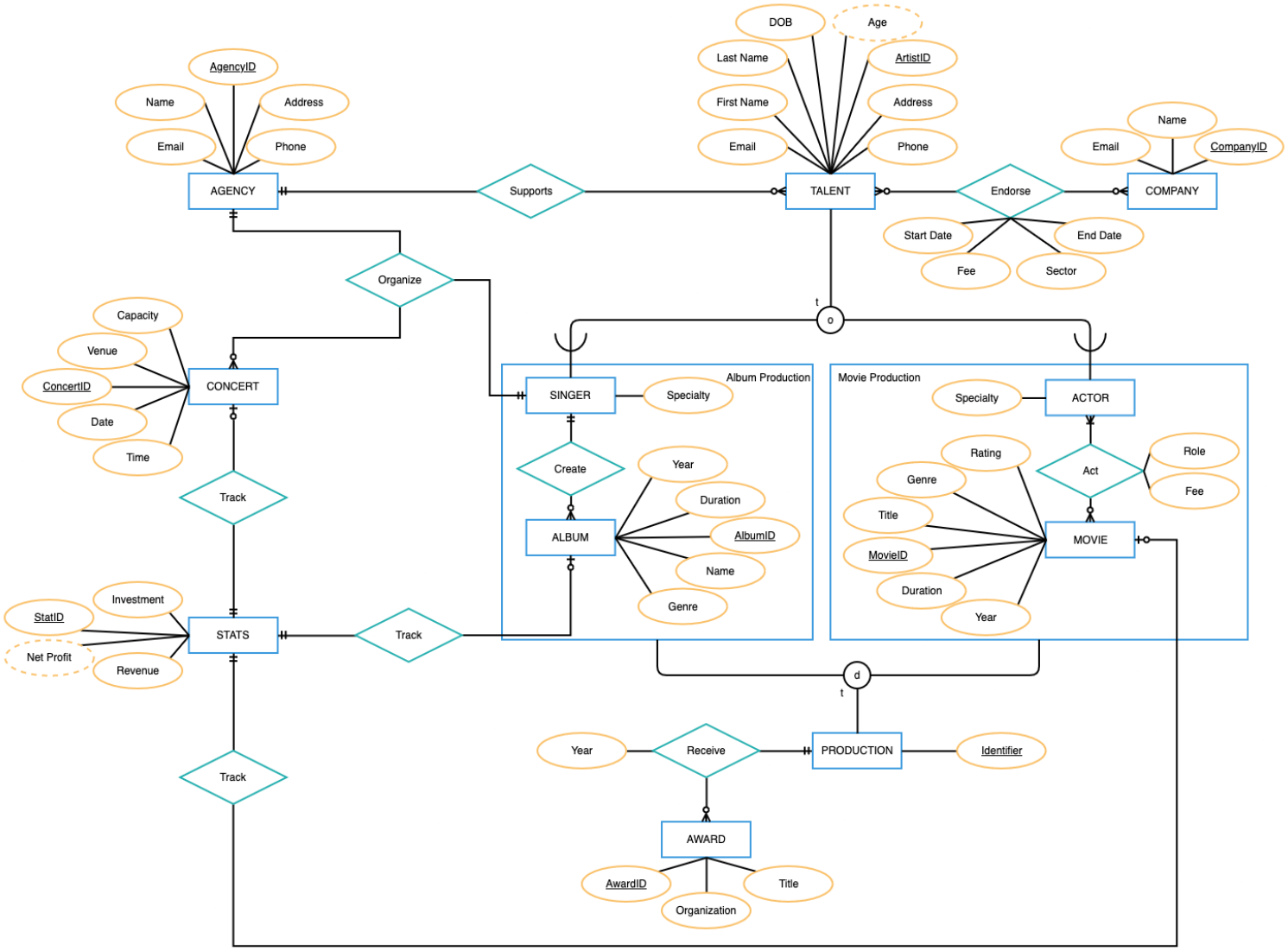
OBJECTIVE

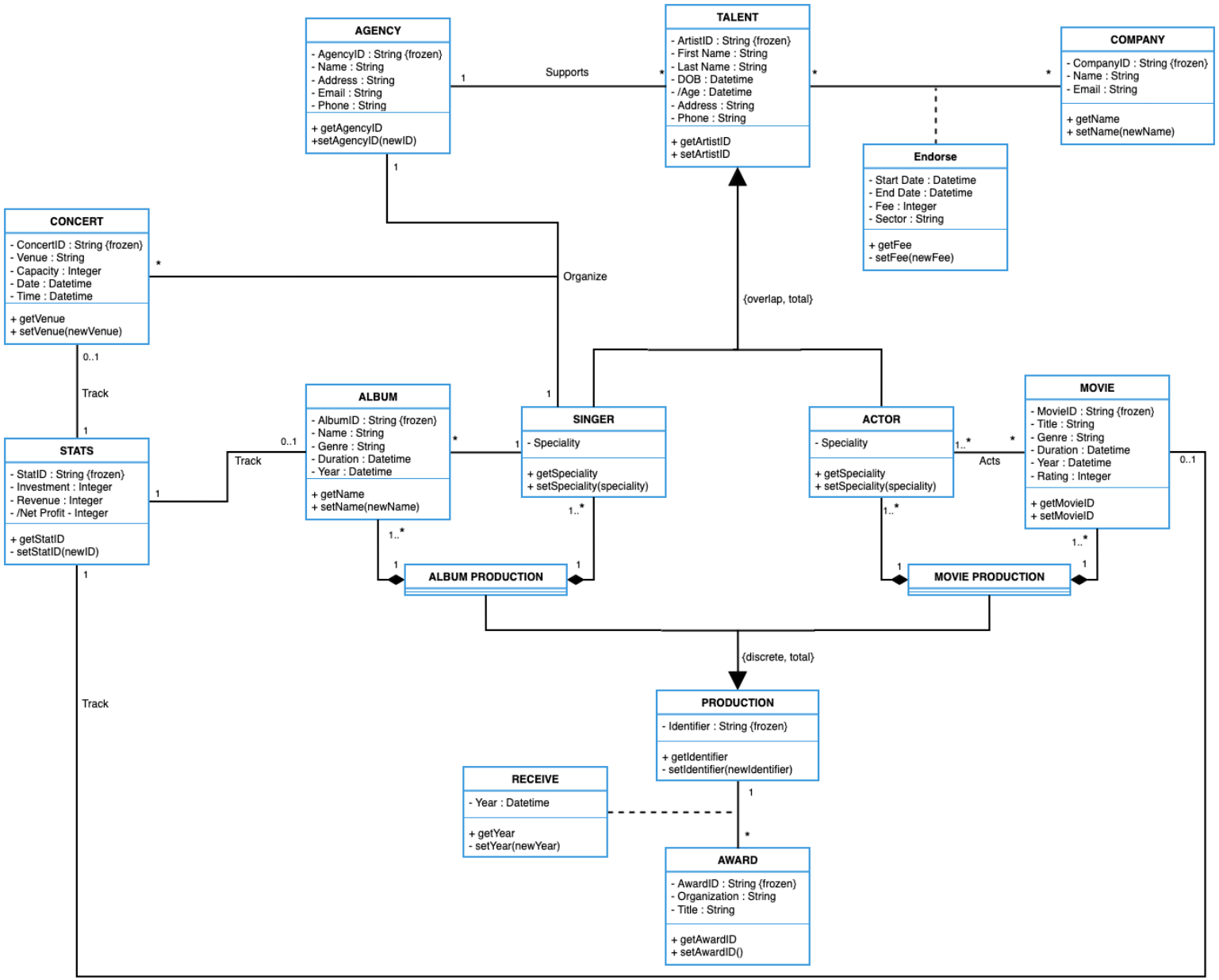
- Centralize data to aggregate and organize diverse data streams for comprehensive insights and analytics.
- Enable informed decisions through advanced data analysis.
- Foster industry evolution with data-driven practices.



DATA MODELS

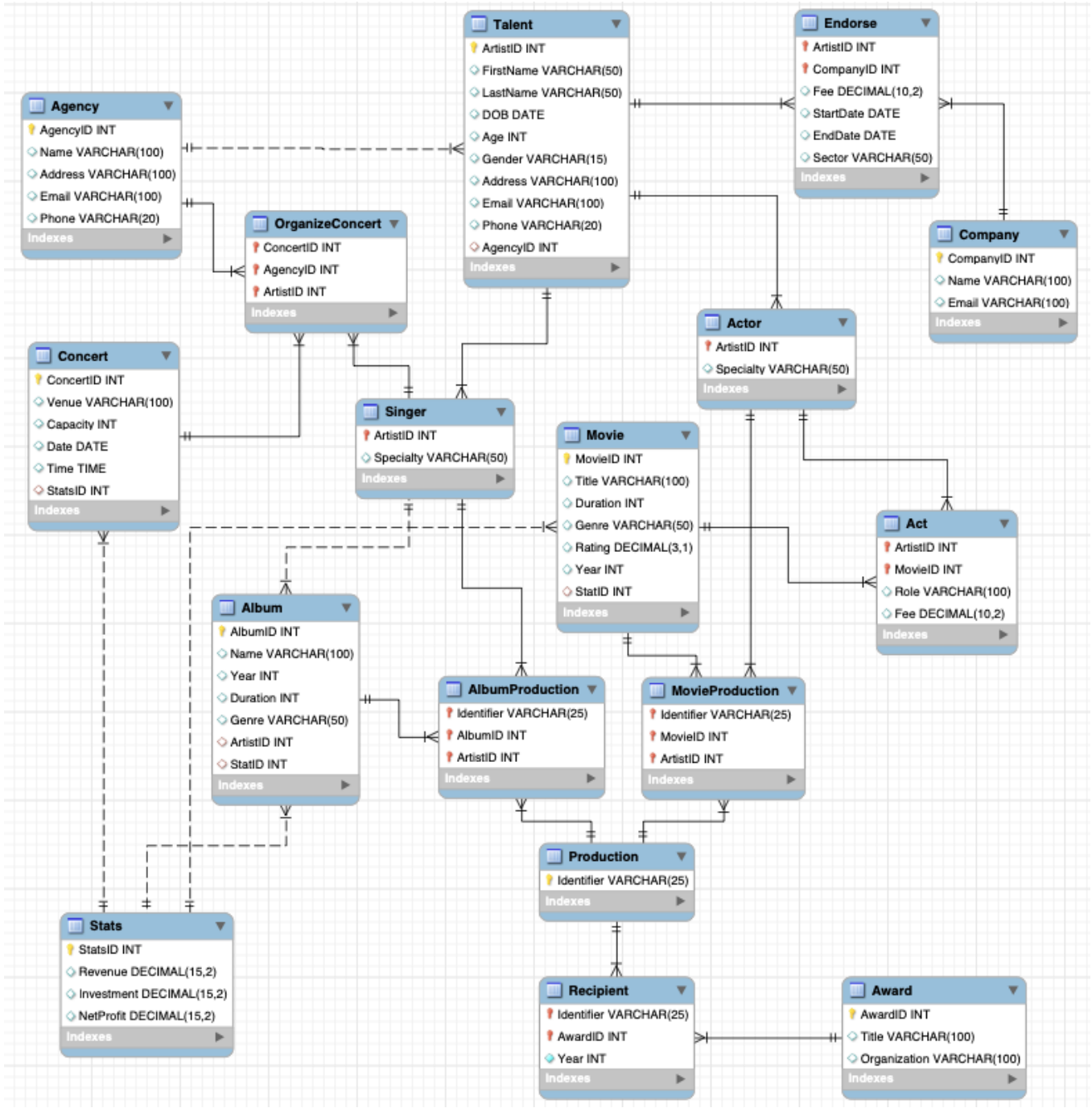
Enhanced Entity-Relationship Diagram




AGENCY
- AgencyID : String {frozen}
- Name : String
- Address : String
- Email : String
- Phone : String
+ getAgencyID
+setAgencyID(newID)
Supports
1
*
TALENT
- ArtistID : String {frozen}
- First Name : String
- Last Name : String
- DOB : Datetime
- /Age : Datetime
- Address : String
- Phone : String
+ getArtistID
+ setArtistID
COMPANY
- CompanyID : String {frozen}
- Name : String
- Email : String
+ getName
+ setName(newName)
Endorse
- Start Date : Datetime
- End Date : Datetime
- Fee : Integer
- Sector : String
+ getFee
- setFee(newFee)
CONCERT
- ConcertID : String {frozen}
- Venue : String
- Capacity : Integer
- Date : Datetime
- Time : Datetime
+ getVenue
+ setVenue(newVenue)
Organize
{overlap, total}
0..1
Track
STATS
- StatID : String {frozen}
- Investment : Integer
- Revenue : Integer
- /Net Profit - Integer
+ getStatID
- setStatID(newID)
ALBUM
- AlbumID : String {frozen}
- Name : String
- Genre : String
- Duration : Datetime
- Year : Datetime
+ getName
+ setName(newName)
SINGER
- Speciality
+ getSpeciality
+ setSpeciality(speciality)
ACTOR
- Speciality
+ getSpeciality
+ setSpeciality(speciality)
Acts
MOVIE
- MovieID : String {frozen}
- Title : String
- Genre : String
- Duration : Datetime
- Year : Datetime
- Rating : Integer
+ getMovieID
+ setMovieID
ALBUM PRODUCTION
MOVIE PRODUCTION
{discrete, total}
PRODUCTION
- Identifier : String {frozen}
+ getIdentifier
- setIdentifier(newIdentifier)
RECEIVE
- Year : Datetime
+ getYear
- setYear(newYear)
AWARD
- AwardID : String {frozen}
- Organization : String
- Title : String
+ getAwardID
+ setAwardID()

UML Diagram

EER Reverse-Engineered



DATA ANALYSIS

Average Revenue Per Concert

USE CASE

An Investors seeks to invest in an agency that organizes concert. Assist in evaluating potential return of investments by indicating efficiency and potential profitability.

INFERENCE

We have identified the Agencies sorted based on average revenue earning per concert. This insight aids in strategic decision-making and maximizing investment returns.

```
19  SELECT a.AgencyID, a.Name, AVG(s.Revenue) AS AvgRevenuePerConcert
20  FROM Concert c
21  JOIN OrganizeConcert oc ON c.ConcertID = oc.ConcertID
22  JOIN Agency a ON oc.AgencyID = a.AgencyID
23  JOIN Stats s ON c.StatsID = s.StatsID
24  GROUP BY a.AgencyID, a.Name
25  ORDER BY AvgRevenuePerConcert DESC
26  LIMIT 10;
27
```

100% 43:30

Result Grid | Filter Rows: Search | Export: | Fetch rows:

AgencyID	Name	AvgRevenuePerConcert
4	Jones-Rohan	236127058.375000
7	Wisoky LLC	232557210.932500
5	Buckridge and Sons	212163483.055000
10	Jast-Strosin	200069349.553333
3	Hegmann-Klein	197064454.515000
6	Muller-Roberts	191012485.320000
1	Reichel, Flatley and Marvin	189207807.085000
8	Gaylord-Block	176890798.012500
2	Von-Walker	168972608.256667
9	Russel PLC	157877050.113333

Revenue Breakdown Over Last 5 Years

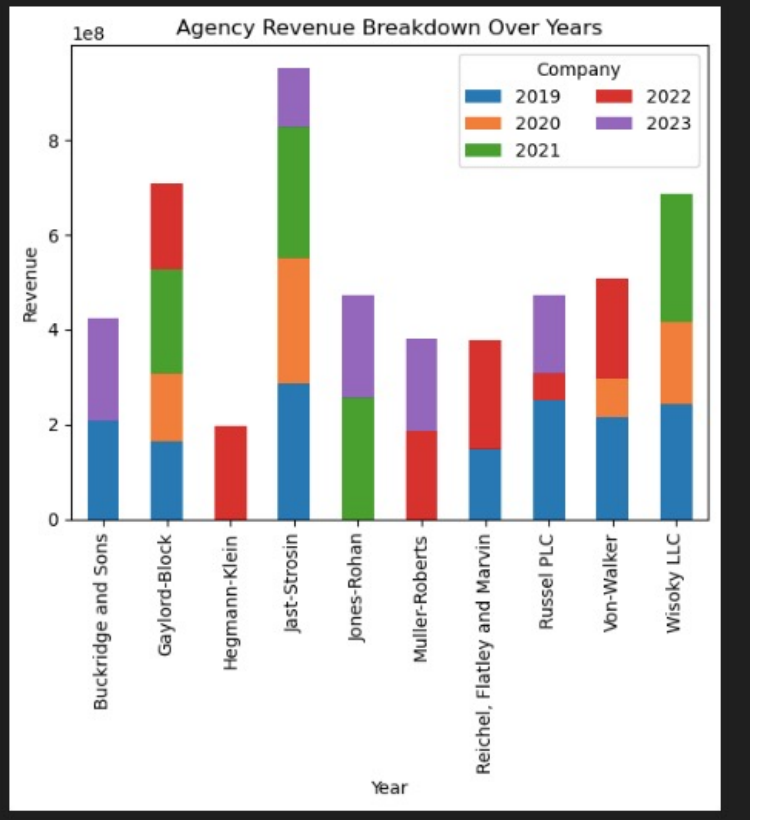
Let's provide additional assistance to the investor by elucidating the revenue generated by each agency over the past five years, along with a detailed breakdown for each year.

```
plt.figure(figsize=(4, 2))
plot1.plot(kind='bar', stacked=True)

plt.title('Agency Revenue Breakdown Over Years')
plt.xlabel('Year')
plt.ylabel('Revenue')

plt.legend(title='Company', loc='upper right', ncol=2)
plt.xticks(rotation=90)
plt.show()
```

```
<Figure size 400x200 with 0 Axes>
```



Top Selling Album Genre

```sql
WITH RankedAlbum AS (
    SELECT
        a.Genre, a.Year, SUM(s.Revenue) AS Revenue,
        RANK() OVER (PARTITION BY a.Year ORDER BY SUM(s.Revenue) DESC) as AlbumRank
    FROM
        Album a,
        Stats s
    WHERE
        a.StatID = s.StatsID
    GROUP BY a.Genre, a.Year
    ORDER BY AlbumRank, Year
)
SELECT *
FROM RankedAlbum
WHERE AlbumRank IN (1,2);
```

100% 55:73

Result Grid Filter Rows: Search Export:

Genre	Year	Revenue	AlbumRank
Pop	2019	55958186.30	1
Pop	2020	36984095.25	1
R&B	2021	50881700.48	1
Classical	2022	43113202.23	1
Pop	2023	63656850.36	1
Party	2019	46944667.14	2
Party	2020	17947783.17	2
Classical	2021	20928505.82	2
Rap	2022	33824691.43	2
Classical	2023	42888435.85	2

USE CASE

An agency aims to understand market demand for various music genres to tailor artist representation and marketing strategies effectively. Help them out.

INFERENCE

We have helped agencies identify the top-selling genres trend in terms of revenue, providing valuable insights into genre-specific consumer preferences. Armed with this information, agencies can make decisions regarding artist signings, promotions, and marketing campaigns to maximize revenue and market share.

Average Age of Artists Represented by Each Agency

USE CASE

An upcoming artist wants to join an agency, eyeing his next career move. He wants to understand the demographics of each agency's talent pool. Let’s help him out.

INFERENCE

With results backing our artist’s decision, he/she can navigate the industry terrain with confidence, selecting the agency best aligned with his artistic vision and career goals.

```
99  SELECT a.AgencyID, a.Name AS AgencyName, AVG(t.Age) AS AverageAge
100 FROM Talent t
101 JOIN Agency a ON t.AgencyID = a.AgencyID
102 GROUP BY a.AgencyID, AgencyName
103 ORDER BY AverageAge;
104
```

100% 29:108

Result Grid Filter Rows: Search Export:

AgencyID	AgencyName	AverageAge
10	Jast-Strosin	16.4000
7	Wisoky LLC	28.2000
3	Hegmann-Klein	35.6000
8	Gaylord-Block	35.8000
5	Buckridge and Sons	36.0000
4	Jones-Rohan	36.4000
6	Muller-Roberts	36.6000
1	Reichel, Flatley and Marvin	37.6000
9	Russel PLC	40.2000
2	Von-Walker	54.2000

Actors' Average Ratings in Movies

```
> db.Act.aggregate([
    { $lookup: {
        from: "Movie",
        localField: "MovieID",
        foreignField: "MovieID",
        as: "movie"
      }
    },
    { $unwind: "$movie" },
    { $group: {
        _id: "$ArtistID",
        avgRating: { $avg: "$movie.Rating" }
      }
    },
    { $sort: { avgRating: -1 } },
    { $limit: 10 }
  ])
< {
    _id: 28,
    avgRating: 9
  }
  {
    _id: 35,
    avgRating: 8.65
  }
  {
    _id: 16,
    avgRating: 8.200000000000001
  }
```

USE CASE

A producer/director wants to cast an actor for their upcoming project. Help them assesses each actors' performance and popularity in movies.

INFERENCE

The output presents each actors’ the average rating of movies they have participated in. The results offer valuable insights for casting decisions, marketing efforts, and strategic planning within the film industry.

What is entertainment without a little gossip?

Entertainment loses its sparkle without the whispers of gossip and the flamboyant displays of our favorite actors.

Let’s dwell into into the enigma of celebrities' financial prowess, adding a layer of intrigue and excitement to the entertainment landscape.

```
140  WITH artistIncome (ArtistID, Income) AS (
141      SELECT t.artistid, COALESCE(a.Fee, 0)
142      FROM talent t
143      LEFT JOIN act a ON a.artistid=t.artistid
144
145      UNION ALL
146      SELECT t.artistid, COALESCE(s.NetProfit, 0)
147      FROM talent t
148      LEFT JOIN album a ON a.artistid=t.artistid
149      LEFT JOIN Stats s ON a.StatID = s.StatsID
150
151      UNION ALL
152      SELECT t.artistid, COALESCE(e.Fee, 0)
153      FROM talent t
154      LEFT JOIN endorse e ON e.artistid=t.artistid
155  )
156  SELECT ai.ArtistID,
157      CONCAT(t.FirstName, ' ', t.LastName) AS ArtistName,
158      SUM(Income) AS netIncome
159  FROM artistIncome ai, Talent t
160  WHERE ai.ArtistID = t.ArtistID
161  GROUP BY ArtistID
162  ORDER BY netIncome DESC;
```

100% 24:132

Result Grid Filter Rows: Search Export:

ArtistID	ArtistName	netIncome
30	Delbert Hayes	208658056.00
31	Beryl Buckridge	187291427.00
19	Lennie Buckridge	170143433.00
50	Russel Mraz	168307863.00
26	Kristian Dooley	162828728.00

THANK YOU