**Chapter 1**

**Introduction**

**1.1 Summary** : The project as the name suggest is a database of information related to films, television programs, including cast, biographies, plot summaries. This aims to provide user with the details of the movie, shows that are being released.

**1.1.1 Usefulness:** Today, the entertainment industry works in millions and the movies are beating all past records in terms of net grossing. This cannot be possible if the industry doesn't have the right ingredient of popularity, awareness among fans. This database serves as the win-win for both the worker behind the scene and for the end-user. It provides the platform to popularise any upcoming tv program /movies , also it gives the user the correct details about the show.

Although there are many such review sites available but we aim to provide user with the list of movies on the type of their genre, favourite show which brings in the new feature in it.

**1.1.2 Realness**: The data that we'll be getting from is by scraping the iMDB website. We have planned to do that by using **IMDbPY** which is a Python package useful to retrieve and manage the data of the IMDb movie database about movies, TV Show, people.

**1.1.3 Motivation:** We call ourselves, “**Stuxnet**”. Everyone of us have a keen interest in movies and TV Shows. Due to this interest we had a fair amount of knowledge as to what kind of data we would be dealing with, where to find it and what the data itself would mean and signify hence it was benefit for us to take up this as our lab project. We were therefore able to sharpen our pre-existing knowledge through this project, getting to know deeper details which otherwise went overlooked.

**Chapter 2**

**DATABASE DETAILS**

**2.1 DATABASE USED**

We used MySQL the world's most popular open source database. With its proven performance, reliability and ease-of-use, MySQL has become the leading database choice for web-based applications, used by high profile web properties including Facebook, Twitter, YouTube, Yahoo! and many more. The database was used by making the use on MySQL workbench used to create databases and the dependencies quiet easily .

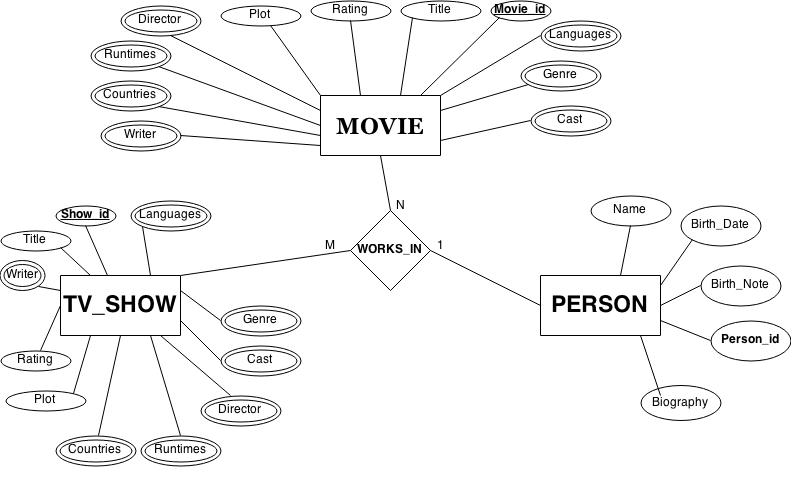
**2.2 E-R DIAGRAM**

**ENTITY INVOLVED**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name Of Entity | Type Of Entity | Attributes | Primary Key | Relationship Involved |
| MOVIE | Strong | Movie\_id,Title,Genre,  Director,Writer,Cast,  Runtime,Country,Language,  Rating,Plot | Movie\_id | Works\_in |
| TV\_SHOW | Strong | Show\_id,Title,Genre,  Director,Writer,Cast,  Runtime,Country,Language,  Rating, Plot | Show\_id | Works\_in |
| PERSON | Strong | Person\_id,Name,Birth\_date,  Birth\_note,Biography | Person\_id | Works\_in |

**RELATIONSHIPS INVOLVED**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name of  Relationship | Entities Involved | Cardinality Ratio | Attributes | Min:Max  Constraints |
| Works\_in | Movie,TV\_Show,  Person |  | None |  |



**2.3 RELATIONAL SCHEMA**

**MOVIE**

|  |  |  |  |
| --- | --- | --- | --- |
| Movie\_id | Title | Rating | Plot |

**TV\_SHOW**

|  |  |  |  |
| --- | --- | --- | --- |
| Show\_ID | Plot | Rating | Title |

**PERSON**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Name | Birth\_Date | Birth\_Note | Person\_id | Biography | Movie\_id | Show\_ID |

**GENRE**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Genre |

**CAST**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Cast |

**RUNTIMES**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Runtimes |

**COUNTRIES**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Countries |

**WRITER**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Writer |

**DIRECTOR**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Director |

**LANGUAGES**

|  |  |  |
| --- | --- | --- |
| Movie\_id | Show\_ID | Languages |

**2.4 FUNCTIONAL DEPENDENCIES**

In each of the tables present in our database the attributes are completely and only dependent on the primary key of the particular table, nothing else dependency is present on any non prime attribute.

**Chapter 3**

**FUNCTIONALITY DETAILS**

**3.1 BASIC FUNCTIONALITY**

|  |  |
| --- | --- |
| Insert | This operation is automated. The movie database gets automatically  inserted in the database. |

**EXCERPTS OF QUERIES:**

**INSERTION OF DATA** :

cursor.execute ("""select \* from Movie where Title=%s""",var)

**3.2 ADVANCED FUNCTIONALITY**

Our code first searches for the user entered Movie or TV show or Person in our database and if not found there then it gets the data from internet from the IMDb library source package in Python. After the data is retrieved, it is automatically inserted in the database for future retrieval.

**EXCERPTS OF QUERIES:**

**WHEN DATA IS NOT IN THE DATABASE IT SEARCHES ONLINE AND INSERTS IN THE DATABASE** :

from imdb import IMDb

ia = IMDb()

for movie in ia.search\_movie(var):

print movie.movieID, movie['title']

id\_movie=raw\_input("\n enter the id of the movie you want to have data of-- ")

inter = ia.get\_movie(id\_movie)

title\_m=genre\_m=director\_m=writer\_m=cast\_m=runtimes\_m=countries\_m=languages\_m=rating\_m=plot\_m=" "

title\_m=str(inter['title'])

print "Title--",title\_m

for genre in inter['genre']:

genre\_m=genre\_m+str(genre)

genre\_m=genre\_m+","

print "Genre--",genre\_m

for cast in inter['cast']:

cast\_m=cast\_m+str(cast)

cast\_m=cast\_m+","

print "Cast--",cast\_m

for runtimes in inter['runtimes']:

runtimes\_m=runtimes\_m+str(runtimes)

runtimes\_m=runtimes\_m+","

print "Runtime--",runtimes\_m

for countries in inter['countries']:

countries\_m=countries\_m+str(countries)

countries\_m=countries\_m+","

print "Countries--",countries\_m

for languages in inter['languages']:

languages\_m=languages\_m+str(languages)

languages\_m=languages\_m+","

print "Languages--",languages\_m

rating\_m=inter['rating']

print "Rating--",rating\_m

for plot in inter['plot']:

plot\_m=plot\_m+str(plot)

print "Plot--",plot\_m

print "\nPopulating"

cursor.execute("""INSERT INTO Movie(Movie\_id,Title,Genre,Director,Writer,Cast,Runtime,Country,Language,Rating,Plot)VALUES(%s,%s,%s,%s,%s,%s,%s,%s,%s,%s,%s);""" ,(id\_movie,title\_m,genre\_m,director\_m,writer\_m,cast\_m,runtimes\_m,countries\_m,languages\_m,rating\_m,plot\_m))

**Chapter 4**

**IMPLEMENTATION DETAILS**

To go about this project, we have gone through various resources to develop our domain, starting from BeautifulSoup to the basics of PYTHON and Tkinter to build GUI which included making of frames, buttons etc. To obtain the legitimate data we had to scrap the data online from IMDb package in PYTHON named IMDbPy.

**4.1 FRONT END**

We’ve used Tkinter.(Python’s de-facto standard GUI) to make our design layout, designing every button and every other widget in a visually appealing manner.

**4.1.1 PYTHON**

Since we needed a programming language whose design philosophy emphasizes code readability, and its syntax allows programmers to express concepts in fewer lines of code than JAVA and C++, Python was our best bet. It integrates with MySql pretty well and creation of UI is possible fairly easy which eased the work for us.

**4.2 BACK END AND FRONT END INTERACTIONS**

MySQL is the relational database management system that we’ve used for our backend and it sinks with Tkinter. It is the world’s most popular open source database and is very effective in managing the database.

A General code Snippet for the connection of the two ends is shown as under:

db = mdb.connect("localhost","amogh","2014","PYMDb" )

cursor = db.cursor()

This is how the back and the front ends get connected and the execution takes place in accordance with both the ends.

**4.2.1 Tkinter and MySQL**

Tkinter gives us command over all the operations we perform at the most basic level, what each click does, what each widget on the screen reacts to or behaves like when the user operates it. It also assists in manipulating and displaying the data accordingly which we obtain via our database. It serves as the backbone for connecting our content with the database on MySQL, allowing us to make queries as per our requirement.

For managing large chunks of data we need to organize them and hence we used MySQL which is an open source relational database management system allowing us to store all the data systematically for easy access and retrieval. MySQL workbench has been used for the development of the database.

**Chapter 5**

**EXPERIENCES**

**5.1 WHAT WE LEARNT**

Database Development is a vast ocean with extra ordinary breadth and depth, which require learning of various aspects at different levels, and hence via this project we were able to explore various areas, which include web scraping, extensive database management, learning new tricks in UI development on Python, handling the windows, linking the buttons, working on the styling and designing part of graphical user interface and connecting our Python code with the database using Tkinter. These were some of the few things we were able to learn due course.

**5.2 PROBLEMS TACKLED**

First and the foremost hurdle for us was how to create a database without manually populating it. After going through our options, we chose to scrap the data from IMDbPy, package in python.

The other problems were a part of our learning aspect was to work with the graphical user interface and the way to link up the data with the database and automatically insert it in the database was a challenge for us but our team managed to tackle it and work efficiently.

**5.3 FUTURE PROSPECTS**

Our current project is a basic/miniature form of what it can actually be, since we have constructed our database with few movies and TV shows. We could mature it in the future by having more content in our database. We would also like to develop our project towards web development.

The main focus in future for our project would be:

* Searching for the movie according to the user’s wish like rating, director, year of release etc.
* Retrieving images along with data from IMDb.
* Creating an android application.

**CONCLUSION**

We hereby conclude that this project on Movie Database gave us an insight of the vast application of Database Management in real. We learned different aspects of backend programming and got aware of the hurdles which helped us gaining experience in this field. We also learned how to work together as a unit and complete our work in time efficiently.

**REFERENCES**

* [www.stackoverflow.com](http://www.stackoverflow.com)
* [www.openbookproject.com](http://www.openbookproject.com)
* [www.tutorialspoint.com](http://www.tutorialspoint.com)
* [www.imdb.com](http://www.imdb.com)
* [www.imdbpy.sourceforge.net](http://www.imdbpy.sourceforge.net)
* [www.zetcode.com/gui/tkinter/](http://www.zetcode.com/gui/tkinter/)
* [www.tkdocs.com](http://www.tkdocs.com)
* [www.python-course.eu/python\_tkinter.php](http://www.python-course.eu/python_tkinter.php)