Data Management Strategy:

Our theater company's software systems are highly dependent on persistent and sensitive data, which is why we have implemented a data management strategy that utilizes a combination of SQL and NoSQL databases to ensure optimal performance, scalability, and security.

Locations Database: Jacob Lorenzo
We are using the NoSQL column store for the locations data base. This is because it allows for easy expansion if more theater locations are added. Each location would be in its own row making the data organized. This also improves the speed of searching through the database because everything is easily accessible. Names of the theater, ID numbers, screen rooms, and location addresses are all information that is kept within the database.

Theaters Database: Jacob Lorenzo
We are using the SQL database because information is likely not going to change. This will ensure that information will be the same at every location. Like the location data base, each theater is allocated its own row. Details pertaining to every theater will be stored. This includes membership status, ID numbers, seat numbers in the screen room. This is basically the locations data base but using SQL instead.

Employee Database:
The Employee Database contains information on who works at which theater and their employee ID's. SQL will allow us to retrieve the data easily into our data management systems.

Users Database: Aeron Flores
The Users database holds vital user information such as ID, name, phone number, email, and address. We have selected a SQL database for this database since there will be a high volume of queries on the database, and SQL databases are designed for these types of queries. We have also logically separated the data so that each user has its row, which enhances the database's performance.

Movies Database: Aeron Flores
The Movies database contains information about movies currently showing, including ID, name, runtime, category, and rating. We have chosen a SQL database for this database due to the dynamic flow of movies that arrive and leave, and it is crucial to ensure that data is consistent and up-to-date for all users. The data has been logically partitioned so that each movie has its own row.

Showtimes Database:

The Showtimes database contains details about the times when each theater will be showing a specific movie. This database contains fields such as Location_ID, Theater_ID, Movie_ID, and Time. We have selected a SQL database for this database because the showtimes can be stored in rows and columns, and the showtimes are set and do not change frequently unless there are fewer or more showtimes available for the particular movie. We have also logically separated the data so that each showtime has its own row.

Diagram:
Here is a diagram that illustrates the structure of our data management strategy:

Locations (NoSQL) -> Theaters (SQL) -> Users (SQL)
Movies (SQL) -> Showtimes (SQL)

Alternative Strategies:
We had considered using a single SQL database to store all of our data, but we rejected the idea due to potential performance issues. We also had the option of using a NoSQL document store for some of our databases, but we concluded that the relational structure of our data is better suited for SQL databases.

Trade Offs: Amara Duru
Our interest in using SQL databases for the majority of the theater system centered around the standardization that their predefined schema allows for, ability to handle transaction-based applications, and guaranteed consistency. Because of the promotion of better performance, scalability, and security provided by SQL databases, we believe that the tradeoffs of additional costs, and the potential need for hardware capacity adjustments down the line, are outweighed by the benefits provided. Another tradeoff is that our data management strategy requires more resources to maintain multiple databases, instead of just one database. However, we have adequate monitoring strategies, have logically separated the data across each database (removing repeating groups of information, promoting reliance on the primary key field, and eliminating general redundancies) to minimize any potential issues with data consistency, and have prepared to maintain relatively stable amounts of data volume to support the function of the databases.