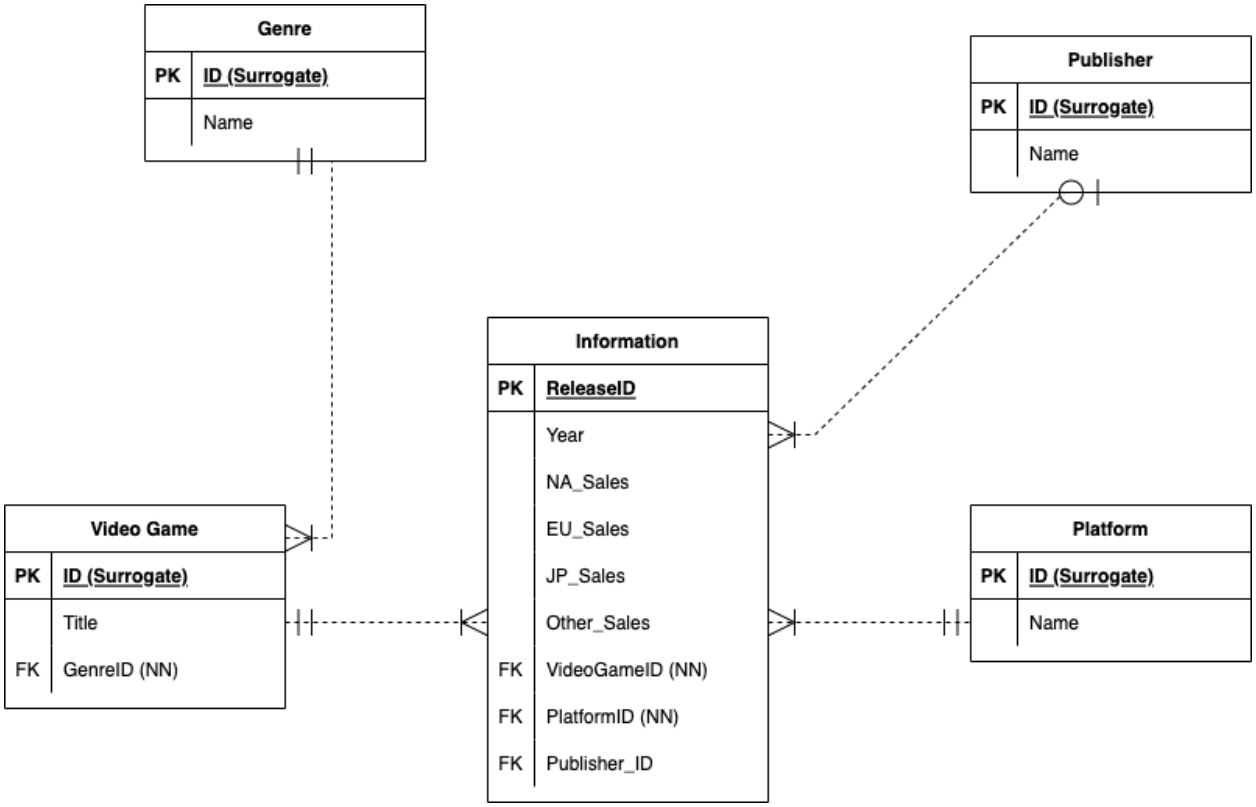
Sean Wendlandt and Matt Krzewinski

11/13/22



Crow's Foot Diagram:

While creating our database design, we had to do several renditions of the crow's foot diagram before deciding on the final model. This was largely due to the discovery of fields we initially thought were unique that we found out were not using queries. For example, one may think that there is only one publisher for each video game title, but that is not the case as it can change when the game is on a different platform. So, many publishers publish many games, and a game can have many publishers. Furthermore, there were some 'N/A' values for the publisher column, so the relationship is optional as it does not need a publisher in the database. After discovering this relationship between publisher and video game was M-M, we knew there would be a linking table somewhere. Similarly, we found that platform and video game was also M-M as the same games are released on many consoles such as playstation, xbox, etc, and these consoles have many games that can be played on them. There were no 'N/A' values for the platform column, so the relationship is mandatory.

We decided to implement one large linking table between these three entities to store all the unique combinations that came up in the data. So, in this table we included Year, the sales attributes, and the surrogate FKs for video game id, publisher id, and platform id. It is worth noting that we decided to drop global sales from the attributes as it is a derived field from the other sales and thus should not be included when designing a database. If we were to include it, and some of the other sales columns were changed, there would need to be a trigger implemented which updated the global sales column or our database would be incorrect. So, it seemed like a much better idea to just drop the column and if global sales is needed, just do a calculation. Similarly, we did not include rank for the same reason. Rank is derived from the global sales of that release, and therefore could frequently change as the sales change. This was by far the largest of all of our tables, which resulted in some migration struggles mainly concerning the foreign keys, which are mentioned below.

The final relationship we have, which was also the only 1-M relationship in the crow's foot diagram, was between Video Game and Genre. This was due to each game only having one genre associated with it in the table, but one genre is associated with many games. This relationship did not change depending on the publisher or platform, so it was best to include a foreign key in our game table. Additionally, there were no 'N/A' values, so the relationship is mandatory.

As we made this design, some of the largest changes came from a further understanding of the attributes of the initial csv table. Specifically, we thought rank was a potential attribute that could serve as a primary key in the linking table as it is unique when combining the game, publisher, and platform. However, since the ranking is based on the sales each game has from their respective platform, additional sales could cause the ranking to change, causing many issues if it is a primary key. This reasoning also caused us to drop ranking from the database altogether as it can be derived. Furthermore, we noted that there weren't any given fields that could serve as primary keys for our tables. Due to this, we used surrogate keys for each table. While we could have created a composite key in the linking table of the game, platform, and publisher id, it seemed easier to just use these as foreign keys for joins.

Migration and Other Issues:

There were also a few issues that came up when trying to migrate the data, primarily with our linking table. However, the first thing to note is that since there were some games that had a publisher of N/A, we did not want a publisher name that was N/A as this is not a real publisher company. So, for our publisher table, we had to select the distinct publishers that were not equal to N/A. While this does cause NULL values in the linking table for pubisher_id, this is more accurate then matching a foreign key for the publisher 'N/A'. The only other initial column with N/As was the year, which is discussed next.

While the sales attributes were able to be inserted with a standard select, year had to be dealt with for a few reasons. First off, a date attribute in mysql requires the format “YYYY-MM-DD,” and the year column in vg_csv was just YYYY, so we had to concatenate the year with “-01-01” to be a valid date entry. Secondly, there were some ‘N/A’ text fields in the vg_csv table so these were not able to be converted to date fields, so we needed to use “CASE WHEN year = 'N/A' THEN NULL ELSE CONCAT(year, '-01-01') END” to be able to handle these instances and get the correct formatting.

The second minor issue we encountered was inserting the foreign keys in the linking table during the bulk insert. While we believed our nested query was correct, our insert would time out and lose connection to the database. However, after some discussions with the professor, we noticed that we needed to increase the time before the query timed out, because the bulk insert due to the size of our data and matching the foreign keys took much more than the default 30 seconds in mysql. In the end, it took just under 200 seconds for this insert, so we certainly needed to increase this default setting.