Advanced System and Data Modeling

System Description

Important Note:

Columns in vg_csv that are to be computed do not need to be listed as attributes (i.e. rank & global sales)

- The ranking is single number that tells where that genre ended up in profit for the region
- Global sales column is simply the sum of all region sales

A **video game** consists of an id, and name

- **Videogame-Release Relationship**
  - *Cardinality*
    - **A game may have multiple releases,** but **a release can only have a single videogame*(1:M)***
  - *Participation*
    - A **game cannot exist without a release** but **a release can exist without a game**
      A game can exist without ever releasing publicly (i.e. "development hell"). On the other hand, a release must have a game associated with it.
      - ***Videogame = optional***
      - ***Release = mandatory***

A **genre** consists of an id and name

- **Genre-Videogame Relationship**
  - *Cardinality*
    - **a genre may belong to multiple games**, but **a game can only have one genre *(1:M)***
      - If the "1-side" of a 1:M relationship is mandatory, FK is NOT NULL
  - *Participation*
    - A **genre can exist without a game**, but **a game cannot exist without a genre**
      Genre serves as a core foundation of a videogame being developed. A video game is just a mechanics simulation as opposed to a "game" .
      - ***Genre = optional***
      - ***Video game = mandatory***
- **Genre-Release Relationship**
  - *Cardinality*
    - **a genre may belong to multiple releases**, but **a release can only have one genre *(1:M)***
      Note: If the "1-side" of a 1:M relationship is mandatory, FK is NOT NULL
  - *Participation*
    - A **genre can exist without a release**, but **a release cannot exist without a genre**
      A genre doesn't need to be part of a game that has released. On the other hand, a release will certainly contain a game that has a genre.
      - ***Genre = optional***
      - ***Release = mandatory***

A **publisher** consists of an id and name

- **Publisher-Release Relationship**
  - *Cardinality*
    - **A publisher may have multiple releases** but **a release can only have a single publisher*(1:M)***
  - *Participation*
    - A **publisher can exist without a release**, but **a release cannot exist without a publisher**
      A publisher does not need to part of a release. On the other hand, a release must have a publisher associated with it.
      - ***Publisher = optional***
      - ***Release = mandatory***

A **platform** consists of an id and name

- **Platform-Release Relationship**
  - *Cardinality*
    - **A platform may have multiple releases,** but **a release will have a single platform*(1:M)***
  - *Participation*
    - A **platform can exist without a release**, but **a release cannot exist without a platform**
      A platform doesn't need to be part of a release(i.e. sales). On the other hand, a release must have a platform associated with it.
      - ***Platform = optional***
      - ***Release = mandatory***

A **Release** consists of na_sales, eu_sales, jp_sales, other_sales, and year

Expected Tables:

1. Genre
2. Videogame
3. Publisher
4. Platform
5. Release (Links Videogame, Platform, and Publisher Tables)

Issues Encountered During Data Migration

My original E-R diagram did not include genre_id as a PK/FK. When writing the stored procedure "addNewReleases" in the file vgStoredProcedures.sql. I kept running into the following error

Error Code: 1054. Unknown column 'genre_id' in 'field list'

when trying to perform the provided call from the assignment description, in the file vgCalls.sql.

call addNewRelease('Foo Attacks', 'X360', 'Strategy', 'Stevenson Studios');

After the addition of a constraint for genre_id in the "releases" table and its corresponding insert statement, of the file vgMigration.sql and without adjusting my stored procedure code post error, I was able to get the expected output message of the new release being added (i.e. calling the stored procedure did not result in the error).

Given the outcome, I have made the necessary modification to my ER Diagram to be submitted.

The Code Discussed Above:

```
-- Create Table vg_releases  - (Adjusted to account for genre_id)
CREATE TABLE vg_releases(release_id BIGINT AUTO_INCREMENT,
                         year DATE,
                         game_id BIGINT,
                         genre_id BIGINT,
                         platform_id BIGINT,
                         publisher_id BIGINT,
                        na_sales FLOAT,
                         eu_sales FLOAT,
                         jp_sales FLOAT,
                         other_sales FLOAT,
CONSTRAINT vg_games_game_id_fk FOREIGN KEY (game_id) REFERENCES vg_games(game_id),
CONSTRAINT vg_genres_genre_id_fk FOREIGN KEY (genre_id) REFERENCES vg_genres(genre_id),
CONSTRAINT vg_platforms_platform_id_fk FOREIGN KEY (platform_id) REFERENCES vg_platforms(platform_id),
CONSTRAINT vg_publishers_publisher_id_fk FOREIGN KEY (publisher_id) REFERENCES vg_publishers(publisher_id),
CONSTRAINT vg_releases_release_id_pk PRIMARY KEY (release_id, game_id, genre_id, platform_id, publisher_id)
);

-- Populate Table vg_releases - (Adjusted to account for genre_id)
-- Note that year in the vg_csv is still a string datatype, need to convert that to date
INSERT INTO vg_releases(release_id, year, game_id, genre_id, platform_id, publisher_id,
                                                    na_sales, eu_sales, jp_sales, other_sales)
SELECT null, IF(year = 'N/A', '0000-00-00', STR_TO_DATE(year, '%Y')),
     game_id, vg_genres.genre_id, platform_id, publisher_id,
            na_sales, eu_sales, jp_sales, other_sales
FROM vg_csv
JOIN vg_games ON vg_games.name = vg_csv.name
JOIN vg_genres ON vg_genres.genre = vg_csv.genre
JOIN vg_platforms ON vg_platforms.platform = vg_csv.platform
JOIN vg_publishers ON vg_publishers.publisher = vg_csv.publisher.
```

```
---------Creating 4th Stored Procedure addNewRelease---
(Un-altered, highlighting what made me think I should include genre_id)
DELIMITER //
CREATE PROCEDURE addNewRelease(
IN gameTitle VARCHAR(1000),
IN platformName VARCHAR(50),
IN genreName VARCHAR(30),
IN publisherName VARCHAR(50)
)
BEGIN
DECLARE platformId BIGINT;
DECLARE genreId BIGINT;
DECLARE publisherId BIGINT;
-- Check if the platform already exists in vg_platforms
SELECT platform_id INTO platformId FROM vg_platforms WHERE platform = platformName;
IF platformId IS NULL THEN
   -- Add new platform to vg_platforms
   INSERT INTO vg_platforms (platform) VALUES (platformName);
   SET platformId = LAST_INSERT_ID();
END IF;
-- Check if the genre already exists in vg_genres
SELECT genre_id INTO genreId FROM vg_genres WHERE genre = genreName;
IF genreId IS NULL THEN
   -- Add new genre to vg_genres
   INSERT INTO vg_genres (genre) VALUES (genreName);
   SET genreId = LAST_INSERT_ID();
END IF;
-- Check if the publisher already exists in vg_publishers
SELECT publisher_id INTO publisherId FROM vg_publishers WHERE publisher = publisherName;
IF publisherId IS NULL THEN
   -- Add new publisher to vg_publishers
   INSERT INTO vg_publishers (publisher) VALUES (publisherName);
   SET publisherId = LAST_INSERT_ID();
END IF;
-- Add new release to vg_releases
INSERT INTO vg_releases (year, game_id, genre_id, platform_id, publisher_id)
SELECT NULL, g.game_id, genreId, platformId, publisherId
FROM vg_games g
WHERE g.name = gameTitle;

SELECT CONCAT('New release added: ', gameTitle, ' on platform ', platformName,
   ' in genre ', genreName, ' by publisher ', publisherName) AS message;
END //
DELIMITER ;
```