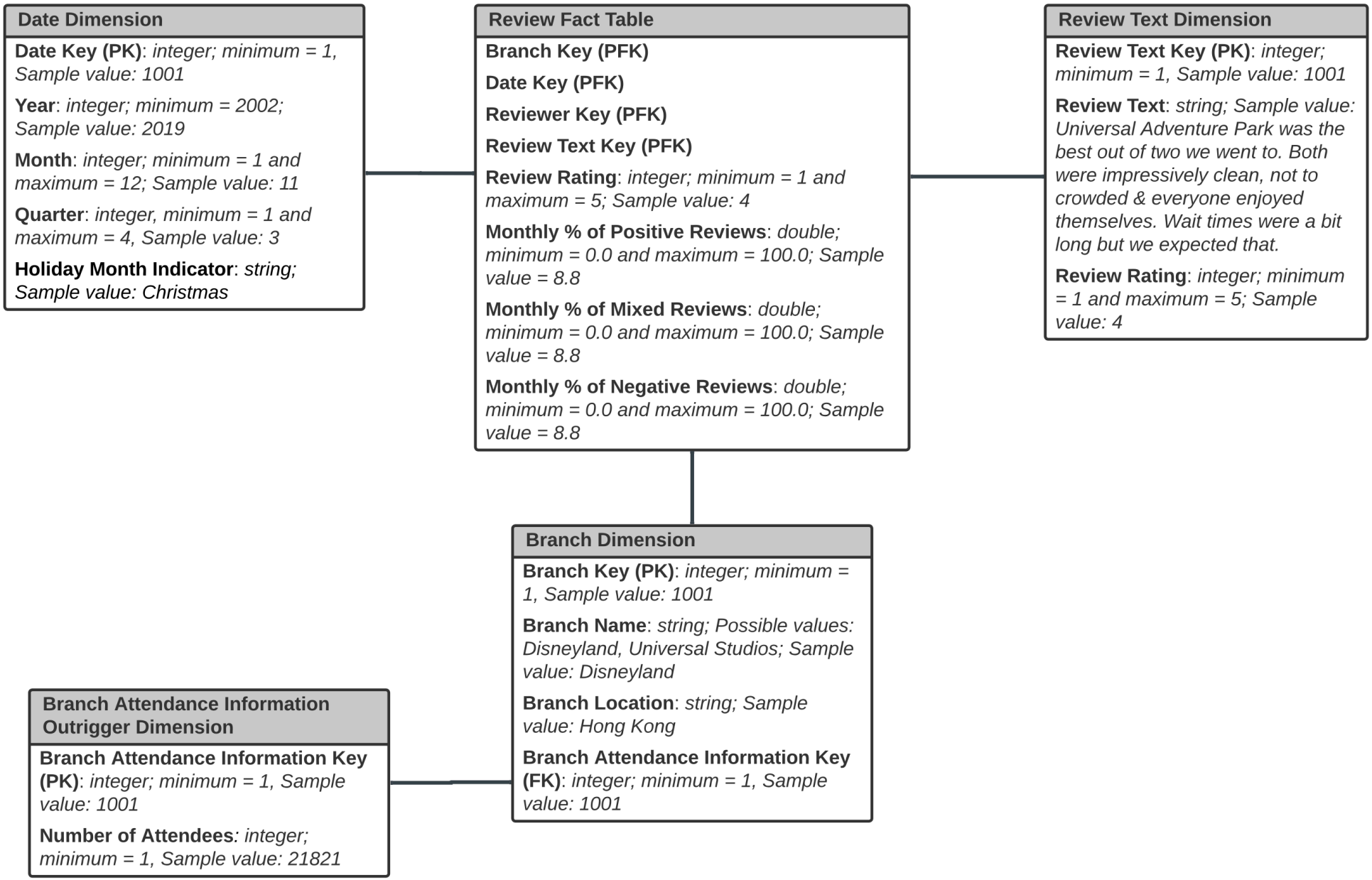
Grain: Monthly rating scaling at Disneyland or Universal Studios for a given branch.



Phase 1 - Conceptual Design

Émilie Brazeau, Nicholas, Gordon Tang

Project plan:

For our project, we intend to enrich the "Disneyland Reviews" dataset with the "Reviews of Universal Studios" dataset. The goal of this project is to build a tool that can predict the rating of a review at a specific Disneyland or Universal Studios branch, given a set of existing reviews with ratings. Our analysis will reveal trends in the writing styles of the different reviews. The dataset contains additional information that can assist this analysis, such as the park branch (Disneyland Paris, Disneyland California, Disneyland Hong Kong, Universal Studios Japan, Universal Studios Florida, or Universal Studios Singapore) and the date the review was written. We can also keep track of the number of attendees per year for each branch (we were unable to locate a freely accessible pre-compiled dataset with attendance information for each park; however, we did find this information for the years 2006-2021 and can manually compile this data - please note, that data can only be found for Universal Studios Singapore beginning in 2010 because that was the year that park opened). Our Disneyland dataset includes reviews from 2010 to 2019, and our Universal Studios dataset includes reviews from 2002 to 2021. We can measure the monthly percentage of positive (4-5 star rated reviews), mixed (3 star rated reviews), and negative (1 star rated reviews) reviews at each branch. We can apply machine learning techniques in later stages of the project to help the analysis.

1. The grain of the data mart

The grain is: Monthly rating scaling at Disneyland or Universal Studios for a given branch.

2. Assumptions

- Reviewers are unable to make changes to their reviews (no reviews in the dataset are edited versions of other reviews).
- All of the reviews are genuine (all reviews were left by actual attendees of a Disneyland or Universal Studios branch; i.e. no reviews were computer-generated).
- We recognize that different people can have the same experience and give it different ratings (for example, two reviewers can write similar reviews, but one reviewer can leave a 2 star rating and the other can leave a 5 star rating). Our analysis should be able to identify outliers in the writing style trends and make predictions accordingly.

3. Checklist of the "10 design mistakes" and how we avoided/handled those mistakes (Where Applicable).

Design Mistake	How We Avoided/Handled Those Mistakes
1. Place text attributes in the Fact table.	To ensure that no text is placed in the Fact table, all text attributes were separated from it. The Review Text, for example, is its own dimension that is linked to the Fact table via a foreign key (not an attribute in the Fact table).
2. Limit verbose descriptions to save space.	Not applicable: We did not limit the length of descriptions.
3. Normalize to save space (leads to slower queries).	We tried to avoid normalizing as much as possible. With the exception of including one outrigger dimension in our schema, we purposefully designed our data mart to adhere to a star schema as much as possible rather than a snowflake schema.
4. Ignore the need to track changes.	Not applicable: We do not anticipate any changes to the data in our data mart (as stated in our assumptions, we assume that reviews in our data mart can not be edited), so we do not need to track changes in the data.
5. Add new hardware to solve all query performance issues.	Not applicable at this stage: We have not encountered any issues with query performance because we are still designing the data mart and have not begun putting queries into action in-system. Because our project includes a machine learning component, we could use the cloud (Google Colab) instead of our own computers to perform the machine learning; this may alleviate future performance issues (not directly related to querying, but to the machine learning) because we are not reliant on the performance of our own computers.
6. Use operational keys as the primary keys.	Not applicable at this stage: When we actually start implementing the data mart (rather than just designing it), we will avoid using operational keys as primary keys.
7. Neglect to declare (and comply with) the grain.	Each dimension was designed with the goal of conforming to the grain.
8. Neglect a detailed	We included all available and relevant dimensions and

design.	attributes to comply with the grain.
9. Expect users to query normalized data. (3 again)	We intentionally designed our data mart to adhere to a star schema rather than a snowflake schema, with the exception of including one outrigger dimension in our schema, so users do not need to query normalized data as much as possible.
10. Fail to conform to Facts and Dimensions.	Due to the nature of our "Disneyland Reviews" and "Reviews of Universal Studios" datasets, we believe it is best to remove attributes that are not shared by the two datasets to avoid conflicts during data integration. In an earlier version of our dimensional model, we had *Day* and *Day of the Week* attributes in the Date Dimension. However, this information is only available in the "Reviews of Universal Studios" dataset; all entries in the "Disneyland Reviews" dataset only have a month and year associated with them (no day). Scaling up the "Disneyland Reviews" data would be impossible because the information about the day (rather than just the month and year) those entries were made is not publicly accessible. *Reviewer Name* and *Review Title* attributes are also associated with entries in the "Reviews of Universal Studios" dataset but not with entries in the "Disneyland Reviews" dataset. The "Disneyland Reviews" dataset has a Reviewer Location associated with it, whereas the "Reviews of Universal Studios" dataset does not. The information required to scale up both datasets so that they match in granularity is also not publicly accessible. As a result, it is best to remove attributes that are not shared by the datasets in order to avoid conflicts during data integration.