Team 4: Aidan Elm, Ryan Farley, Phil Kershner

CSC 351 - Database Management Systems

9 Dec 24

Dr. Forouraghi

**DBMS Final Project**

***Digital Lego Cataloging***

**Description and Functionalities**

Our group designed and created a user-friendly website for cataloging, searching, and customizing LEGO sets and their pieces. Users are given the ability to view LEGO sets, observe single pieces, and set up their personal LEGO sets of their own. Sets are fully customizable, with users having the ability to add and remove pieces, both to existing and created sets.

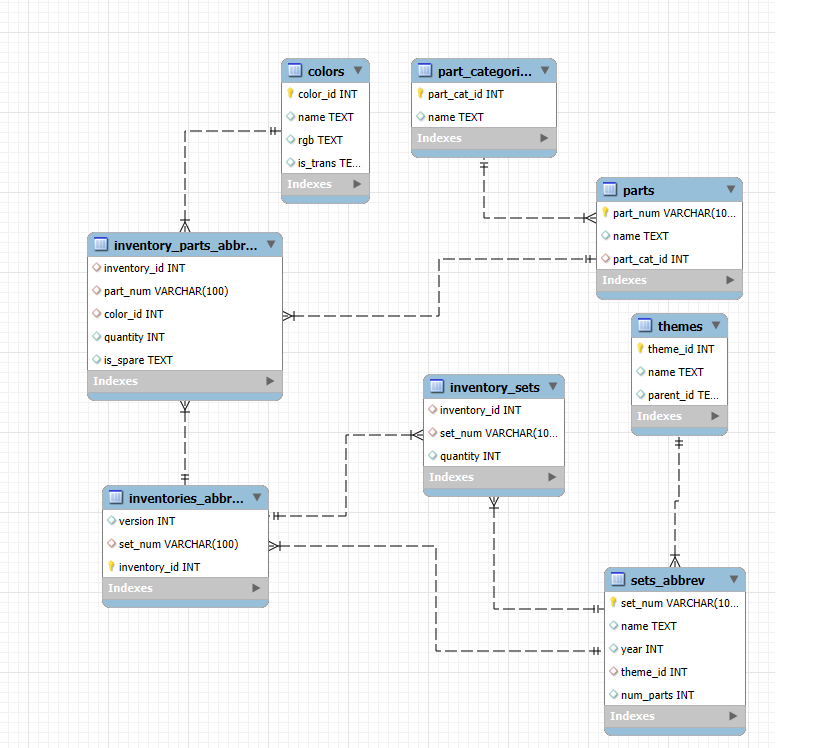
The project utilizes a 3-tier client/server application with MySql, NodeJS, and an html website. Sets and pieces are stored at the lowest tier of the system within the MySql database. The middle level is our two NodeJS files, which enable connection between the top and bottom levels and transfer information between them. The highest level, where users interact with our program, is our html website Brickbase.

**Functionalities**

* Sql functionalities:
  + INSERT INTO: Users can insert pieces into their custom sets.
  + DELETE: Users can delete pieces from their custom sets if they are no longer needed.
  + SELECT: Retrieves and displays stored custom sets for the user.
  + UPDATE: Modify existing custom sets: add or remove pieces.
* LEGO Sets Research: Specifications of a variety of LEGO sets can be viewed and searched.
* View Individual Pieces organized by LEGO part number.
* Custom set creation: Users can pick which pieces make up their unique sets.

**Files and descriptions**

* app.js
  + The main server-side script for a Node.js web application. It implements an Express-based web server to manage a LEGO database, providing functionality for users to view sets, search for parts and sets, create custom sets, and manage parts within those sets. The script also includes validation and error-handling mechanisms to ensure input safety and proper database interactions.
* db\_connection.js
  + A module for establishing a connection between the Node.js web server and the MySQL database LegoDB. It uses the mysql2 library to configure and initiate the database connection with specific credentials. The module exports the database connection object to enable database queries throughout the application.
* legodb.sql
  + A structured SQL script for the initialization, modification, and querying of the legodb database. It defines database schema constraints, establishes relationships through primary and foreign keys, and includes operations for managing and querying LEGO sets and parts data. The script also includes specific search queries for iconic sets like the "Death Star" and "Porsche 911," showcasing its utility for both data setup and retrieval in the project.
* index.html
  + Serves as the primary entry point for the BrickBase web application. It is designed to provide a visually appealing and functional interface for users to explore LEGO bricks and sets, create custom sets, and manage LEGO parts. Additional CSS, JS, and asset files provide styling to this file.
* main.css
  + Defines the global and component-specific styling for the BrickBase project, ensuring consistency and a visually appealing user interface.

**Schema**

**Statement of Ranking**

*Member 1: Phil Kershner*

My teammates and I agree that I handled 34% of the overall project. My specific tasks included:

Task 1: Team leader, advocating to Dr. Forouraghi for the group with our proposal.

Task 2: Importing the tables into MySQL and making smaller versions of tables due to size limitations using pandas in python.

Task 3: Initialized tables with keys and data type constraints and created associated MySQL queries for our SELECT, DELETE, INSERT, and CREATE functions. Created schema.

Task 4: With the help of Ryan, connected MySQL to Node JS “middleman.”

*Member 2: Aidan Elm*

My teammates and I agree that I handled 33% of the overall project. My specific tasks included:

Task 1: Created the front-end website HTML + CSS with the Bootstrap framework.

Task 2: Connected the front-end site to the Node JS "middleman."

Task 3: Assisted in the creation of the project research + documents.

*Member 3: Ryan Farley*

My teammates and I agree that I handled 33% of the overall project. My specific tasks included:

Task 1: I focused on the Node Js layer of the project.

Task 2: With Phil's assistance restructured the backend into two js files (Started with several, tried to use just one, settled on two) : app.js & db\_connection.js

Task 3: Populated app.js with the ability to select, insert into, update and/or delete sql with get and post methods.

Task 4: Assisted in project research and document creation.

\*Task 5: Provided entertainment to the group by texting messages of confusion anytime I was lost and had no idea what I was doing.

\*Not an actual task