Schema

Project (ProjectID, ProjectName, ProjectAuthor, ProjectDescription, ProjectCreationDate)

ProjectComponent (ProjectID, ItemID, Quantity)

- Foreign Key ProjectID references Project
- Foreign Key ItemID references Item

Item (ItemID, Namespace, BlockID, ItemName, IsRaw, IsSmeltable, IsBlastable, IsSmokable, BurnTime)

Recipe (RecipeID, ResultItemID, ResultQuantity, RecipeType, RecipeImage)

- Foreign Key ResultItemID references Item

RecipeIngredient (RecipeID, ItemID, Quantity)

- Foreign Key RecipeID references Recipe
- Foreign Key ItemID references Item

Tag (TagID, Namespace, TagName)

TagMember (TagID, ItemID)

- Foreign Key TagID references Tag
- Foreign Key ItemID references Item

Table Explanations

- Project
    - Description: The Project table represents Minecraft projects created by users. A project is a build that the user intends to create in one or more Minecraft worlds. This table provides some basic information about the project, some of which is used by the ProjectComponent table to specify the items required for the project. This table can answer questions like “How many total projects have been created?”, “How many different authors have created a project?”, and “Which author has created the most projects?”
    - Columns:
        - ProjectID – primary key (auto number)
        - ProjectName – the name of the project; assigned by the project author
        - ProjectAuthor – the user who created the project
        - ProjectDescription – an optional description of the project; may be NULL
        - ProjectCreationDate – the date on which the project was created
- ProjectComponent
    - Description: The ProjectComponent table allows a list of components to be specified for each project in the Project table. Each row represents one of the items required to build a project by identifying the project it is a part of and the item it represents through references to the Project and Item tables. This table can answer questions like “What do I need for Project X?” and “How many projects use item Y in their recipe?”
    - Columns:
        - ProjectID – foreign key; the ID of the project to which the row belongs; combines with ItemID to form the primary key
        - ItemID – foreign key; the item for which the row is specifying a quantity; combines with ProjectID to form the primary key
        - Quantity – the quantity of the specified item to be used in the project

- Item
    - Description: The Item table contains basic information about each item available in Minecraft. This table can answer questions like “What items are available to use?”, “Which items are raw resources?”, and “Which items are smeltable?”
    - Columns:
        - ItemID – primary key (auto number)
        - Namespace – the namespace in which this item is defined; for vanilla Minecraft, the value here is “minecraft”
        - BlockID – the ID used by the Minecraft code base to uniquely identify an item within a namespace; for example, a vanilla diamond pickaxe has the ID “diamond_pickaxe”
        - ItemName – the name of the item, as it is displayed to players in the game
        - IsRaw – Boolean; indicates whether the item is a raw resource or must be crafted
        - IsSmeltable – Boolean; indicates whether or not the item can be smelted in a furnace
        - IsBlastable – Boolean; indicates whether or not the item can be smelted in a blast furnace
        - IsSmokable – Boolean; indicates whether or not the item can be smelted in a smoker
        - BurnTime – the time (in game ticks) that an item burns for when used as fuel in a furnace; a value of 0 indicates that an item cannot be used as fuel; example: one piece of coal has a burn time of 1600 ticks

- Recipe
    - Description: The Recipe table contains basic information about the various recipes defined in Minecraft. Each row references the Item table to specify the result of the operation, and a reference image of the in-game crafting interface is provided to give visual context in the application interface. This table can answer questions like “How many different ways are there to craft item X?” and “How many smelting recipes are there?”
    - Columns:
        - RecipeID – primary key (auto number)
        - ResultItemID – foreign key; identifies the item that results from this recipe
        - ResultQuantity – the number of the resulting item that is created by this recipe
        - RecipeType – indicates what kind of recipe this is; acceptable values: “crafting” and “smelting”; acceptable values may be expanded in the future to support crafting methods introduced by mods
        - RecipeImage – the URL to an image showing the in-game recipe
- RecipeIngredient
    - Description: The RecipeIngredient allows for a list of ingredients and their quantities to be defined for each recipe in the Recipe table. Each ingredient identifies the recipe to which it belongs through a reference to the Recipe table and which item it represents though a reference to the Item table. Those two foreign keys combine to form a composite primary key for this table. This table can answer questions like “How many unique ingredients are in recipe X?” and “How many different recipes use item Y?”
    - Columns:
        - RecipeID – foreign key; the ID of the recipe to which the row belongs; combines with Namespace and ItemID to form the primary key
        - ItemID – foreign key; the item for which the row is specifying a quantity; combines with RecipeID to form the primary key
        - Quantity – the quantity of the specified item that is required for the specified recipe

- Tag
    - Description: A table containing basic information about the tags used by Minecraft to identify groups of similar blocks that can replace each other in recipes and other game functions.
    - Columns:
        - TagID – primary key (auto number)
        - Namespace – the namespace in which this item is defined; for vanilla Minecraft, the value here is “minecraft”
        - TagName – the name of the tag; for example, in vanilla Minecraft, all plank variants (oak, spruce, acacia, dark oak, birch, and jungle) are grouped into one tag called “planks”
- TagMember
    - Description: TagMember links an item to any tag(s) it is a part of.
    - Columns:
        - TagID – the ID of the tag the item belongs to
        - ItemID – the ID of the item that belongs to the tag

Evidence of Normalization

- First Normal Form: Each column in every table may contain only one value at a time. Where it was possible to have multiple values at a time in a single column, such columns have been pulled out into their own tables (ProjectComponent and RecipeIngredient).
- Second Normal Form:
    - The Project and Recipe tables do not have composite keys; therefore, they cannot violate 2NF.
    - In the Item table, all columns are functionally dependent on both the Namespace and the ItemID because an item can only be uniquely identified in Minecraft using both of them, thus all other attributes depend on both parts of the key and the Item table does not violate 2NF.
    - In the ProjectComponent and RecipeIngredient tables, different projects or recipes may use differing quantities of the same item, thus we must know both the project or recipe **and** the item in order to accurately specify the quantity, therefore, these tables do not violate 2NF.
- Third Normal Form:
    - Each table contains only one function dependency, as shown below, thus showing that each table is in 3NF. Each FD is defined in the form "(Table) FD".
        - (Project) FD: ProjectID → ProjectName, ProjectAuthor, ProjectDescription
        - (ProjectComponent) FD: (ProjectID, Namespace, ItemID) → Quantity
        - (Item) FD: (Namespace, ItemID) → ItemName, IsRaw, IsSmeltable, IsBlastable, IsSmokable, FuelValue
        - (Recipe) FD: RecipeID → Namespace, ItemID, RecipeType, RecipeImage
        - (RecipeIngredient) FD: (RecipeID, Namespace, ItemID) → Quantity
- Fourth Normal Form: We solved our problem with the First Normal Form by placing all multi-valued attributes into their own tables (ProjectComponent and RecipeIngredient), which means our database does not violate Fourth Normal Form.