**How do I query the Inventory?**

1. **Get All Items in an Aquarium:** To retrieve all items associated with a specific aquarium, you can join the Inventory table with the respective reference tables based on the entity\_type and entity\_id.

#SQL Statement:

*SELECT entity\_type, entity\_id*

*FROM Inventory*

*WHERE aquarium\_id = ?;*

Replace ? with the specific aquarium\_id you're interested in.

1. **Get All Fish in an Aquarium**: To get details of all fish associated with a specific aquarium:.

#SQL Statement:

*SELECT Fish.\**

*FROM Inventory*

*JOIN Fish ON Inventory.entity\_id = Fish.id*

*WHERE Inventory.aquarium\_id = ? AND Inventory.entity\_type = 'Fish';*

1. **Get All Plants in an Aquarium**:

#SQL Statement:

*SELECT Plant.\**

*FROM Inventory*

*JOIN Plant ON Inventory.entity\_id = Plant.id*

*WHERE Inventory.aquarium\_id = ? AND Inventory.entity\_type = 'Plant';*

1. **Get All Equipment in an Aquarium**:

#SQL Statement:

*SELECT Equipment.\**

*FROM Inventory*

*JOIN Equipment ON Inventory.entity\_id = Equipment.id*

*WHERE Inventory.aquarium\_id = ? AND Inventory.entity\_type = 'Equipment';*

1. **Get the Date an Item was Added to an Aquarium:** If you want to know when a specific item (e.g., a fish) was added to an aquarium:

#SQL Statement:

*SELECT date\_added*

*FROM Inventory*

*WHERE aquarium\_id = ? AND entity\_type = 'Fish' AND entity\_id = ?;*

Replace the first ? with the aquarium\_id and the second ? with the specific fish's id.

1. **Get the Date an Item was Added to an Aquarium:** To retrieve all items added to an aquarium on a particular date:

#SQL Statement:

*SELECT entity\_type, entity\_id*

*FROM Inventory*

*WHERE aquarium\_id = ? AND date\_added = ?;*

Replace the first ? with the aquarium\_id and the second ? with the desired date.

When executing these queries in Python using SQLite, you can use the cursor.execute() method and pass the parameters as a tuple. For example:

#Python:

aquarium\_id = 1

cursor.execute("SELECT entity\_type, entity\_id FROM Inventory WHERE aquarium\_id = ?", (aquarium\_id,))

results = cursor.fetchall()

**Add more entity types**

Yes, you can add more entity types to the Inventory table. However, there are a few steps you should follow to ensure the integrity and functionality of your database:

1. **Update the Database Schema:**

Before adding new entity types, you should update the database schema to include tables for these new entities. For example, if you want to add a new entity type called **"Decorations"**, you should have a Decorations table in your database schema that contains all the relevant columns for decorations.

1. **Update the ‘Inventory Table’:**

The **‘entity\_type’** column in the **‘Inventory’** table should be updated to include the new entity types. This can be done by modifying the CHECK constraint or by using an ENUM type (if your database supports it). For SQLite, you'd modify the CHECK constraint.

For example, to add "Decorations" and "Medications" as new entity types:

#SQL Statement:

*ALTER TABLE Inventory*

*ADD CONSTRAINT chk\_entity\_type CHECK(entity\_type IN ('Fish', 'Plant', 'Equipment', 'Decorations', 'Medications'));*

1. **Update Application Logic**: If you have application logic (in Python or any other language) that interacts with the database, ensure that it's updated to handle the new entity types. This includes CRUD operations, validations, and any other relevant logic.
2. **Data Migration (if necessary):** If you're adding new entity types based on existing data, you might need to perform data migration. This involves moving data from existing tables or external sources to the new tables you've created.
3. **Testing:** After making these changes, it's crucial to test your application thoroughly. Ensure that all functionalities related to the new entity types work as expected and that there are no issues with the existing functionalities.
4. **Backup:** Before making any significant changes to your database, always take a backup. This ensures that you can restore your database to its previous state in case anything goes wrong.