Using Django and PostgreSQL for your web application, you’ll want to design your database schema to reflect the different entities involved in managing gas stations (branches, staff, shifts, sales, inventory, etc.). Here are some key entities (tables) and their attributes based on your project description:

1. Branch (Outlet)

Represents each gas station branch.

Table: ‘Branch’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘name’ (String, max 100)

- ‘location’ (String, max 255)

- ‘manager\_id’ (Foreign Key to ‘User’ table, Integer) – Refers to the manager in charge of the branch.

- ‘created\_at’ (Timestamp) – When the branch was added.

- ‘updated\_at’ (Timestamp) – Last update to the branch record.

2. User (Staff)

Represents users in the system, including managers, supervisors, and pump attendants. Use Django’s User model for authentication and extend it for additional fields.

Table: ‘User’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘username’ (String, unique)

- ‘password’ (Hashed, handled by Django’s auth)

- ‘role’ (String, choices: [Manager, Supervisor, Attendant]) – User role to handle permissions.

- ‘branch\_id’ (Foreign Key to ‘Branch’) – The branch the user is assigned to.

- ‘first\_name’ (String, max 100)

- ‘last\_name’ (String, max 100)

- ‘email’ (String, unique)

- ‘created\_at’ (Timestamp)

- ‘updated\_at’ (Timestamp)

3. Shift

Represents a work shift for pump attendants (morning and afternoon shifts).

Table: ‘Shift’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘name’ (String, max 50) – Morning Shift, Afternoon Shift.

- ‘start\_time’ (Time) – Shift start time.

- ‘end\_time’ (Time) – Shift end time.

4. AttendantShift

Tracks which pump attendant is working during a specific shift.

Table: ‘AttendantShift’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘attendant\_id’ (Foreign Key to ‘User’) – The pump attendant.

- ‘shift\_id’ (Foreign Key to ‘Shift’) – The shift they worked on.

- ‘branch\_id’ (Foreign Key to ‘Branch’) – Branch where the shift was worked.

- ‘date’ (Date) – Date of the shift.

- ‘inventory\_start’ (Decimal) – Amount of fuel inventory at the start of the shift.

- ‘inventory\_end’ (Decimal) – Amount of fuel inventory at the end of the shift.

5. Inventory

Represents the inventory of fuel at each branch, which is managed by the supervisor.

Table: ‘Inventory’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘branch\_id’ (Foreign Key to ‘Branch’)

- ‘fuel\_type’ (String, choices: [Petrol, Diesel, etc.]) – Type of fuel.

- ‘quantity’ (Decimal) – Current quantity of fuel in stock.

- ‘last\_updated’ (Timestamp)

6. Sale

Represents sales transactions recorded by pump attendants. Each sale is associated with a shift.

Table: ‘Sale’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘attendant\_shift\_id’ (Foreign Key to ‘AttendantShift’) – Attendant who made the sale.

- ‘branch\_id’ (Foreign Key to ‘Branch’)

- ‘fuel\_type’ (String, choices: [Petrol, Diesel, etc.])

- ‘quantity\_sold’ (Decimal) – Amount of fuel sold.

- ‘total\_price’ (Decimal) – Total price of the sale.

- ‘sale\_time’ (Timestamp)

7. Reconciliation

Tracks reconciliations performed by supervisors to ensure sales match inventory changes.

Table: ‘Reconciliation’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘branch\_id’ (Foreign Key to ‘Branch’)

- ‘supervisor\_id’ (Foreign Key to ‘User’) – Supervisor performing reconciliation.

- ‘attendant\_shift\_id’ (Foreign Key to ‘AttendantShift’) – Shift being reconciled.

- ‘total\_sales\_quantity’ (Decimal) – Total quantity sold during the shift.

- ‘inventory\_recorded’ (Decimal) – Quantity of inventory at the end of the shift.

- ‘reconciliation\_time’ (Timestamp)

8. Report

Stores reports generated by managers.

Table: ‘Report’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘branch\_id’ (Foreign Key to ‘Branch’)

- ‘manager\_id’ (Foreign Key to ‘User’)

- ‘report\_type’ (String, choices: [Daily, Weekly, Monthly, Quarterly, Annual])

- ‘start\_date’ (Date) – Start date of the reporting period.

- ‘end\_date’ (Date) – End date of the reporting period.

- ‘total\_sales’ (Decimal) – Total sales for the period.

- ‘total\_inventory\_used’ (Decimal) – Total inventory used during the period.

- ‘generated\_at’ (Timestamp)

9. ConsolidatedReport

Consolidated report for the MD, combining data from multiple branches.

Table: ‘ConsolidatedReport’

- ‘id’ (Primary Key, Integer, Auto-increment)

- ‘md\_id’ (Foreign Key to ‘User’) – The MD generating the report.

- ‘report\_type’ (String, choices: [Daily, Weekly, Monthly, Quarterly, Annual])

- ‘start\_date’ (Date)

- ‘end\_date’ (Date)

- ‘total\_sales\_all\_branches’ (Decimal) – Combined total sales for all branches.

- ‘total\_inventory\_used\_all\_branches’ (Decimal) – Combined total inventory used.

- ‘generated\_at’ (Timestamp)

Relations Between Tables:

- User → Branch: Each branch has a manager, supervisor, and attendants.

- Shift → AttendantShift: Tracks which attendants worked which shifts.

- AttendantShift → Sale: Links sales to specific shifts.

- Branch → Inventory: Each branch has its own fuel inventory.

- Reconciliation: Supervisors reconcile sales and inventory for each shift.

- Reports: Managers generate reports for their branches, and the MD consolidates reports.

Notes:

- Use Django’s ORM (Object-Relational Mapping) to manage these tables. Django will generate the corresponding SQL for PostgreSQL, making it easy to interact with the database through Django models.

- Utilize Django's admin panel for managing these tables in a user-friendly way.

This schema should cover your basic needs for managing sales, inventory, users, shifts, and reporting. Let me know if you'd like further refinements or if you have specific use cases that need more details!