Design-1

Specific roles and users will be created for each team.

The users are only allowed access to certain materialized views and not the underlying tables.

Logistics Team:

* Team should have access only to have access to the sales data
* Create a materialized view with a query to retrieve the transaction data, filtering to only the transaction ID, items bought, total weight and, total price
  + Should the team require the data to be updated frequently
    - Create a trigger function to refresh the materialized view after inserts, updates, or deletes to the base table
  + Should the team be satisfied with delayed data
    - Scheduling a job (Cronjob or Airflow) to access the database and trigger the refresh
  + Disadvantages:
    - Manual or scheduled refreshing
    - Stale data depending on refresh frequency
    - During refresh, view is blocked, preventing queries
      * Can be avoided using CONCURRENTLY during view creation
* Team should not update the table directly to modify the status of the transactions
  + A separate application/API should be used to authenticate and verify the team members
  + After verification, the application/API will submit the update query based on the transaction ID provided by the user

Analytics Team:

* Create materialized views of the queries, allowing the team access to only specific parts of the data
* Materialized views should also prevent the team members from updating the tables
* For analytics, would recommend using an OLAP database which is optimized for analytics workloads

Sales Team:

* Should not be able to access the database/tables directly
* Create a separate application/API to authenticate/verify the team members and, managed the update operations
* Delete operations should also be performed via the application/API