A short description of the final project, and what it accomplished

Our project, named “Kalahari” is a representation of a warehouse database. We used HTML and Javascript using the JQuery library for frontend and SQL with PostgreSQL for backend. The server is supported by node.js.

It allows three types of users: customer, company, and admin. Customers can browse through items on catalog, filter items based on their id, and purchase items. Companies can add items and view a list of items that the company has added to the warehouse. Admin can see the list of unshipped orders from customers, and decide to approve or reject the order. If the orders are approved, admins can see the list and statistics. Admins can also delete existing warehouses and move all items there to another warehouse location.

A description of how your final schema differed from the schema you turned in.

If the final schema differed, why?

- We added in **cost** (double) as a attribute to an item because we thought it would be a crucial information for customers to see before purchasing an item. The cost can also help an admin see the performances of the warehouses, by seeing the maximum or minimum average cost of all items in warehouses.
- We removed **req_num** from items because for our warehouse implementation we realized that item has no dependency on shipping request.
- We added a **quantity** field to Shipping Request in order to keep track of the number of items a user can add to their cart before checking out.
- We also added a a **shipped** flag to Shipping Request to give admin the power to approve or reject an order that a customer places. If the order is approved, the flag will indicate that order should appear in the shipped order tab. If the order is rejected, then the order would be discarded.

1. Item(I_ID, weight, quantity, cost, volume, **lat**, **lon**, **ID**)
2. Warehouse(lat, lon, capacity)
3. Shipping_Req(req_num, qty, origin, dest, total_val, shipped, **veh_ID**, **ID**, **lat**, **lon, I_ID**)
4. User(ID)
5. Customer(cu_name, **ID**)
6. Company(co_name, **ID**)
7. Shipping_Method(veh_ID)
8. Air(air_dist_mult, **veh_ID**, air_base_cost)
9. Land(land_dist_mult, **veh_ID**, land_base_cost)
10. Sea(sea_dist_mult, **veh_ID**, sea_base_cost)

A list of all SQL queries used

"SELECT * FROM ITEM WHERE I_ID= $1"

"SELECT * FROM ITEM"

“select * from item where ID = $1"

“SELECT * FROM WAREHOUSE"

“SELECT * FROM SHIPPING_REQUEST WHERE shipped=0"

"SELECT * FROM SHIPPING_REQUEST WHERE shipped=1"

select I_ID, qty from shipping_request where req_num = $1"

select quantity from item where I_ID = $1"

"update shipping_request set shipped=1 where req_num=$1"

"update item set quantity=$1 where I_ID=$2"

"delete from shipping_request where req_num = $1"

SELECT max(req_num) FROM shipping_request

SELECT * FROM SHIPPING_METHOD WHERE veh_id = $1

SELECT * FROM USERS WHERE id = $1

SELECT * FROM WAREHOUSE WHERE lat = $1 AND lon = $2

SELECT * FROM ITEM WHERE i_id = $1

INSERT INTO SHIPPING_REQUEST VALUES($1, $2, $3, $4, $5, $6, $7, $8, $9, $10, $11

SELECT * FROM SHIPPING_REQUEST WHERE ID = $1

SELECT * FROM SHIPPING_METHOD

SELECT id FROM customer WHERE cu_name = $1

SELECT id FROM company WHERE co_name = $1

SELECT max(i_id) FROM ITEM

SELECT * FROM WAREHOUSE WHERE lat = $1 AND lon = $2

SELECT * FROM COMPANY WHERE ID = $1

INSERT INTO ITEM VALUES($1, $2, $3, $4, $5, $6, $7, $8)

SELECT * FROM SHIPPING_REQUEST WHERE ID = $1 AND shipped = 0

SELECT * FROM SHIPPING_REQUEST WHERE ID = $1 AND shipped = 1

SELECT i_id, count(req_num) FROM shipping_request GROUP BY i_id ORDER BY count(req_num) DESC

DELETE FROM USERS WHERE id = $1

SELECT cu.* FROM customer cu WHERE NOT EXISTS((SELECT co.id FROM company co) EXCEPT(SELECT i.id FROM item i, shipping_request sr WHEREsr.i_id = i.i_id AND cu.id = sr.id))

select MAX(groups.av) from (select avg(cost) as av, item.lat, item.lon from item group by lat, lon) as groups

select MAX(groups.av) from (select avg(cost) as av, item.lat, item.lon from item group by lat, lon) as groups

SELECT * FROM ITEM WHERE lat = $1 AND lon = $2

UPDATE ITEM SET lat = $1, lon = $2 WHERE lat = $3 AND lon = $4

DELETE FROM warehouse WHERE lat = $1 AND lon = $2

A list of all functional dependencies that are applicable to the table (including the ones involving the primary key). For each functional dependency, briefly describe its meaning in English.

1. I_ID -> Weight, quantity, Cost, Volume, lat , lon ,req_num, ID
   Explanation: Represents an item in a warehouse

2. lat, lon -> capacity
   Explanation: Represents a Warehouse

3. req_num -> qty, origin, dest, total_val, shipped, veh_ID, ID, lat, lon. I_ID
   Explanation: Represents a shipping request made by a customer.

4. ID -> ID (trivial)
   Explanation: Represents a user, which is either a Customer or a Company

5. ID -> cu_name
   Explanation: Represents a customer

6. ID -> co_name
   Explanation: Represents a company

7. veh_ID -> veh_ID (trivial)
   Explanation: Represents a shipping method, which is either an Air, Land or Sea transport.

8. veh_ID -> air_dist_mult, air_base_cost
   Explanation: Represents an Air transport (and cost)

9. veh_ID -> land_dist_mult, land_base_cost
   Explanation: Represents an Land transport (and cost)

10. veh_ID -> sea_dist_mult, sea_base_cost

Explanation: Represents an Sea transport (and cost)