Muhammad Haseeb 20100192
Arbab Arshad
Naseer Alam

I have done complete work (SQL QUESRIES + FUNCTIONAL FRONT-END OF ALL OF THEM).

SQL QUERIES:
***I have also *optimized* most of the queries.**

#**Query 1**
```
SELECT pet_type, COUNT(pet_type) AS total_sales
FROM (SELECT * FROM sale_main LEFT JOIN pets_main ON sale_main.product_id = pets_main.pet_id) as JT
GROUP BY pet_type
ORDER BY COUNT(pet_type) DESC
LIMIT 1;
```

#**Query 2**
```
SELECT stp.supplier_id, COUNT(stp.supplier_id) AS total_sales
FROM supplier_to_prodcut as stp
GROUP BY stp.supplier_id
ORDER BY COUNT(stp.supplier_id) DESC
LIMIT 1;
```

#**Query 3**
```
SELECT SUM(pet_price) as Revenue, sale_date
FROM (SELECT * FROM sale_main LEFT JOIN pets_main ON sale_main.product_id = pets_main.pet_id) as
RESD
WHERE sale_date = 20181111;
```

#**Query 4**
```
SELECT cm.customer_id, cm.customer_name, cm.customer_city, cm.customer_street,
RESD.total_amount_of_purchases FROM
customer_main as cm
RIGHT JOIN
(SELECT customer_id as customer, SUM(pet_price) as total_amount_of_purchases
FROM sale_main as sm LEFT JOIN pets_main as pm
ON sm.product_id = pm.pet_id
GROUP BY customer_id
ORDER BY SUM(pet_price) DESC
LIMIT 1) as RESD
ON cm.customer_id = RESD.customer
```

#**Query 5**
```
SELECT fm.food_name as leastEatenFoodName, (fm.food_id) as leastEatenFoodID
FROM food_main as fm LEFT JOIN pet_to_food as ptf
ON fm.food_id = ptf.food_id
GROUP BY fm.food_id
ORDER BY COUNT(fm.food_id)
LIMIT 1
```

#**Query 6**
```
SELECT RESD.customer_id, RESD.sale_id, RESD.pet_id, RESD.pet_name, RESD.pet_type, RESD.pet_price,
RESD.sale_date, RESD.employee_id
FROM (SELECT * FROM sale_main as sm
LEFT JOIN pets_main as pm ON sm.product_id = pm.pet_id) as RESD
WHERE RESD.customer_id = 1 # VAR
```

#**Query 7**
```
SELECT pet_details.PID as ID, pet_details.PN as NAME, pet_details.PT as Type, pet_details.PB as Breed,
pet_details.PG as Gender, pet_details.PSize as Size, pet_details.PP as Price
FROM sale_main as sm LEFT JOIN
(SELECT pm.pet_id as PID, pm.pet_name as PN, pm.pet_type as PT, ps.pet_breed as PB, ps.pet_gender as PG,
ps.pet_size as PSize, pm.pet_price as PP
FROM pets_main as pm LEFT JOIN pets_specs as ps ON pm.pet_id = ps.pet_id) as pet_details
ON sm.product_id = pet_details.PID
WHERE sm.sale_id = 2 # VAR
```

#**Query 8a**
```
SELECT COUNT(product_id) as TOTAL_SALES, supplier_id FROM supplier_to_prodcut
WHERE supplier_id = 1
ORDER BY COUNT(product_id)
```

#**Query 8b**
```
SELECT stp.supplier_id as SUPPLIER, pm.pet_name as NAME, pm.pet_type TYPE FROM
(supplier_to_prodcut as stp LEFT JOIN pets_main as pm ON stp.product_id = pm.pet_id)
WHERE stp.supplier_id = 1 #VAR
```

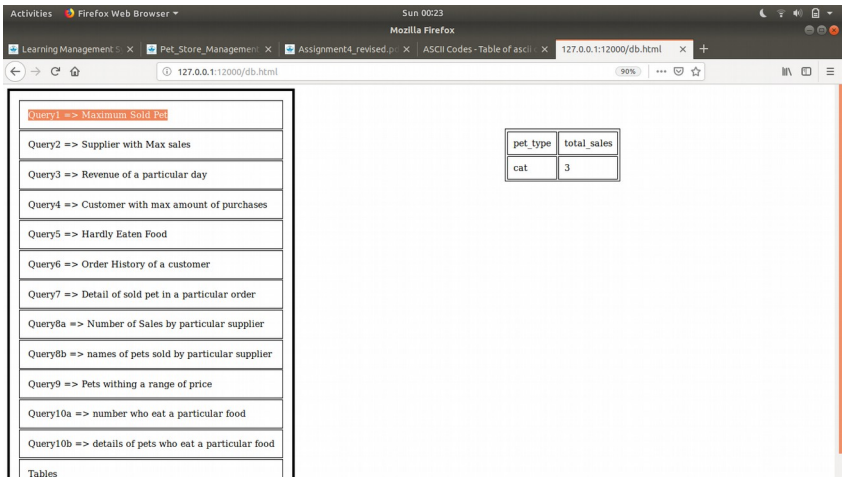
#**Query 9**
```
SELECT pet_name FROM pets_main
WHERE pet_price > 1 AND pet_price < 100
```

#**Query 10a**
```
SELECT COUNT(RESD.PET)
FROM (SELECT ptf.pet_id as PET, ptf.food_id as FOOD FROM pet_to_food as ptf WHERE food_id = 2) as
RESD
```

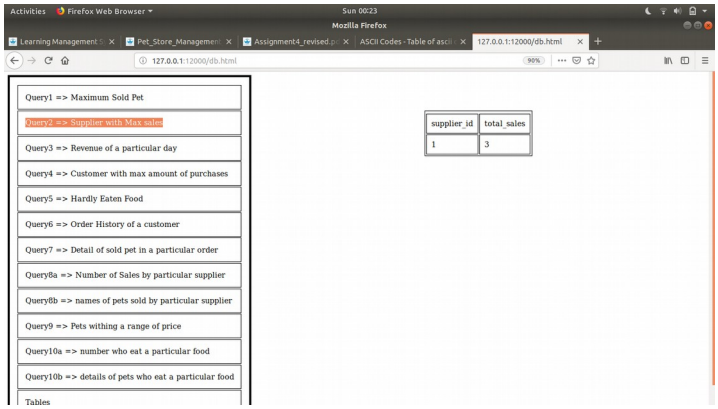
#**Query 10b**
```
SELECT pet_details.PID, pet_details.PT, pet_details.PB, pet_details.PG, pet_details.PSize, pet_details.PP FROM
pet_to_food as gptf LEFT JOIN
(SELECT pm.pet_id as PID, pm.pet_name as PN, pm.pet_type as PT, ps.pet_breed as PB, ps.pet_gender as PG,
ps.pet_size as PSize, pm.pet_price as PP
FROM pets_main as pm LEFT JOIN pets_specs as ps ON pm.pet_id = ps.pet_id) as pet_details ON gptf.pet_id =
pet_details.PID
WHERE gptf.food_id = 2 # VAR
```

Screen-shots of all ten queries' functional front-end:

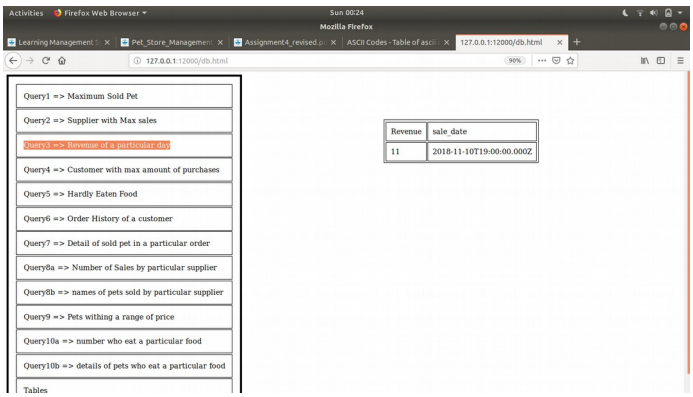
Query1:

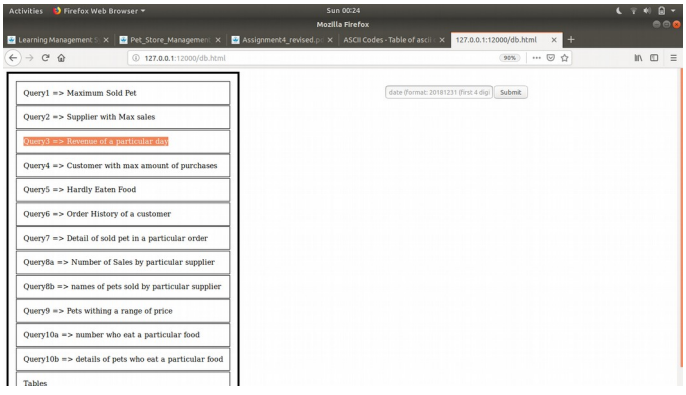


Query2:

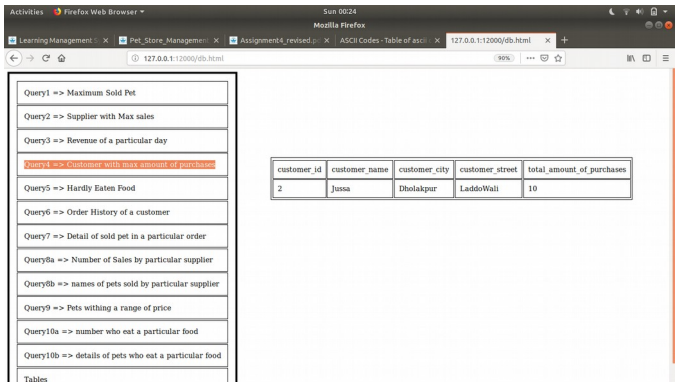


Query3:

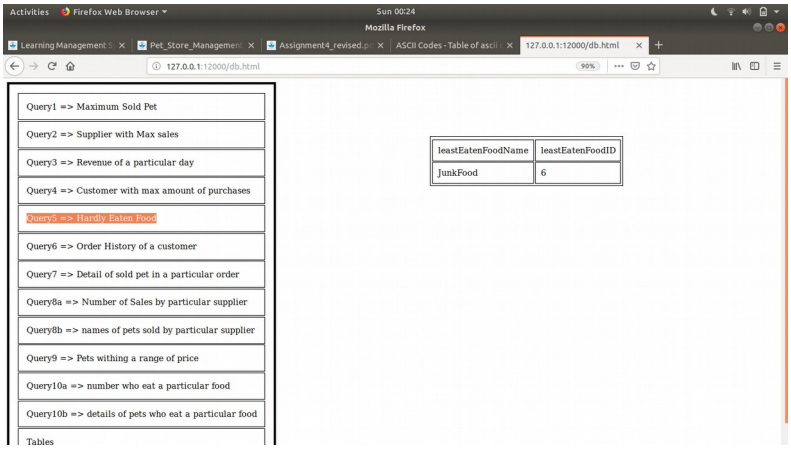




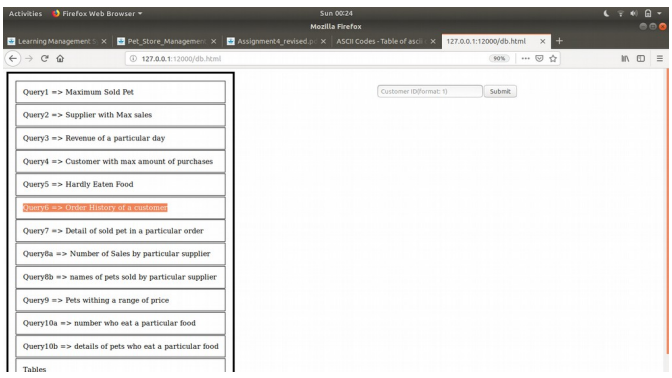
Query4:

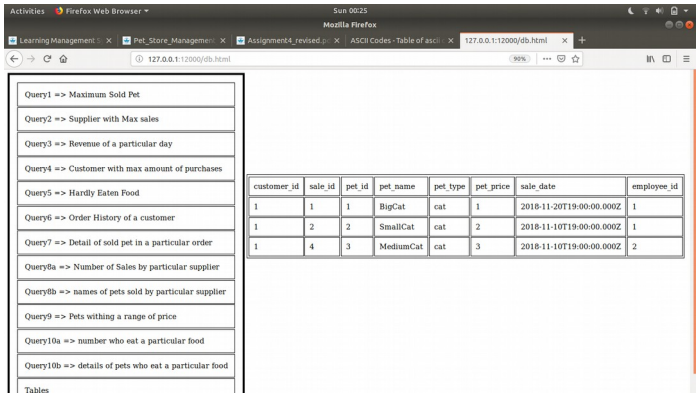


Query5:

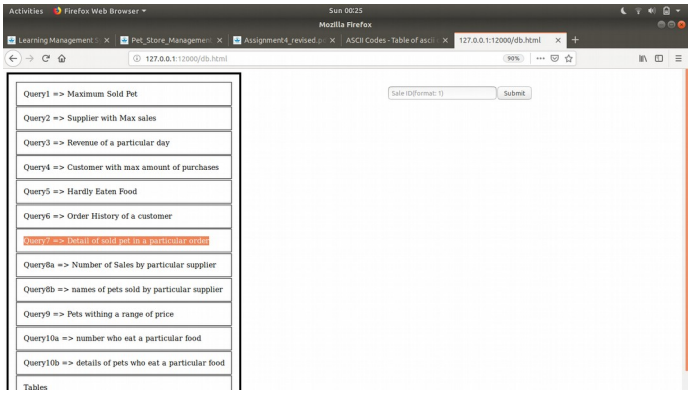


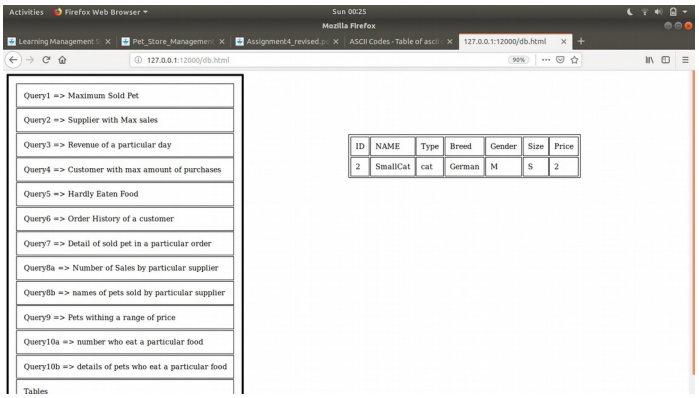
Query6:



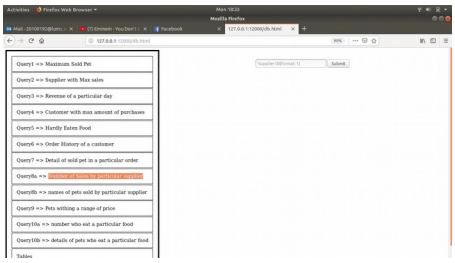


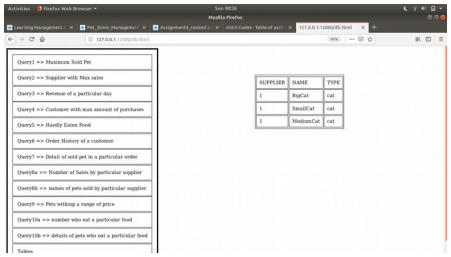
Query7:

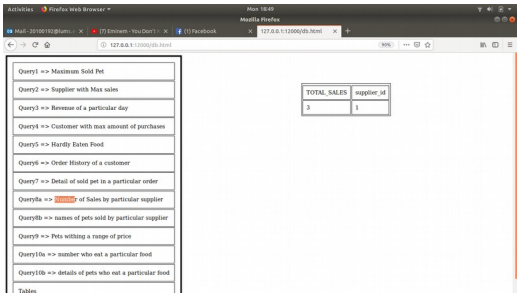




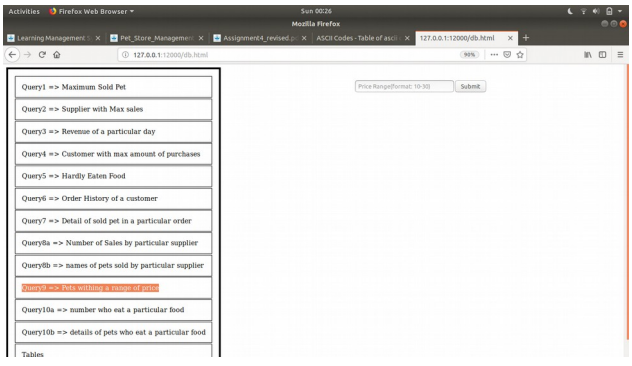
Query8:

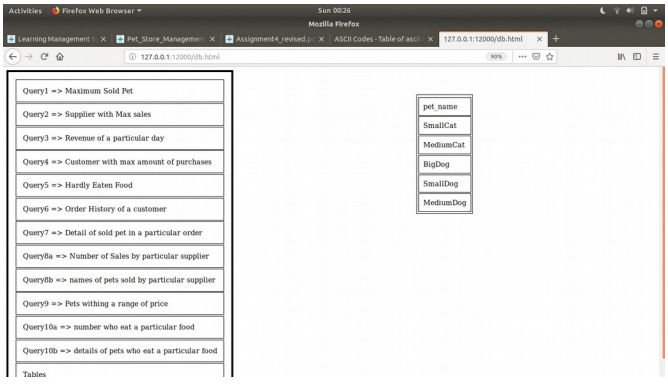




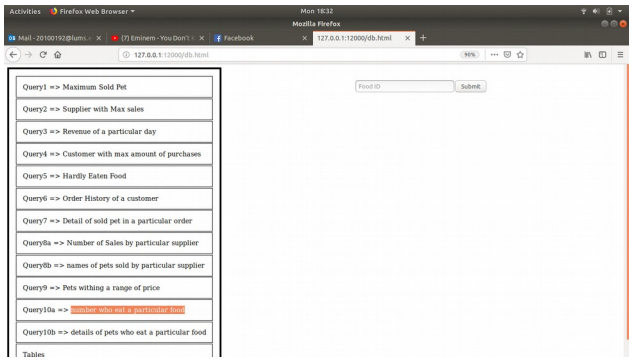


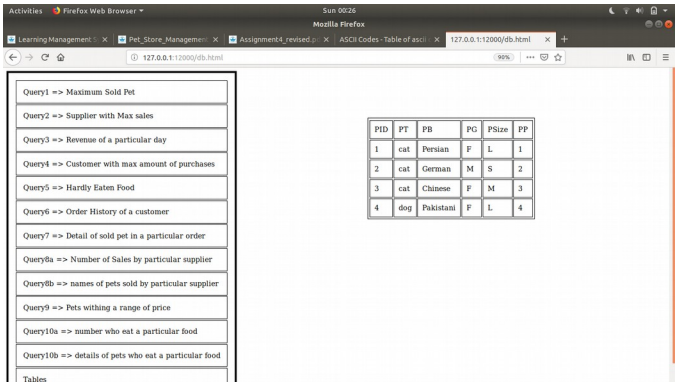
Query9:





Query10:





All Done!
Note: Front-end may not be beautiful but it is fully functional and most of the queries are *optimized*.