Deliverable II
**Starts After Page 3
Pet Store Inventory Database

Team
Jeanie Handler - Diagrams, Database
Geoffrey Greenleaf - Project Planning/Coordination, Diagrams, Dataset

Introduction

Our pet store inventory database stores relevant information in order to run a simple pet store where a customer can buy pets, pet food, and pet toys. The database stores pet store information which includes employees as well as information on pet products. We chose the pet store topic because Jeanie loves animals. Ideally working with this subject will resolve the issue of learning more about databases.

Comparison

In our design for the pet store we have 3 entities and 3 relationships. The three entities are Pet Store, Employee, and Product, and the relations are transaction, stocks, employs. The relationships are as follows. A Store employs an employee, A Store also stocks a product. and a A Store sells a product. In the example given by relationaldbdesign.com(1) In the example there are 3 entities as well, a customer, product, and pet_care_log. There is a table for the relationship between customer and item sold. A difference between our product and the product of the example is the example also has a package which is recursively defined as multiple products. The pet log table will handle information if the customer requests pet care as one of the products. In our database we do not have the option for pet care we mainly focus on selling different variations of pet products such as fish tank, dog bowls, collars, etc… In our schema we also don't store customer information because we don't mind not having the customer information for every product sold. In the future maybe a customer table could be added that would allow for repeat customers to have a discounts off some products but for now no information about the customer is stored. The example also only focuses on a single pet store in our example we have many pet stores because it could be considered a national chain and we want to know which store has certain items in stock in case the customer needs to order from a website instead of going to the store. In the example the sale_item table is similar to our sells table in that we both have foreign keys to the product but instead of a key to a customer we have a key to the store which the product came from. Each transaction will have a unique transaction id for that transaction. 1 Store can have multiple transactions but there can not be multiple transactions per store.

1. http://www.relationaldbdesign.com/programming-pl-sql/module1/database-pet-store-schema.php

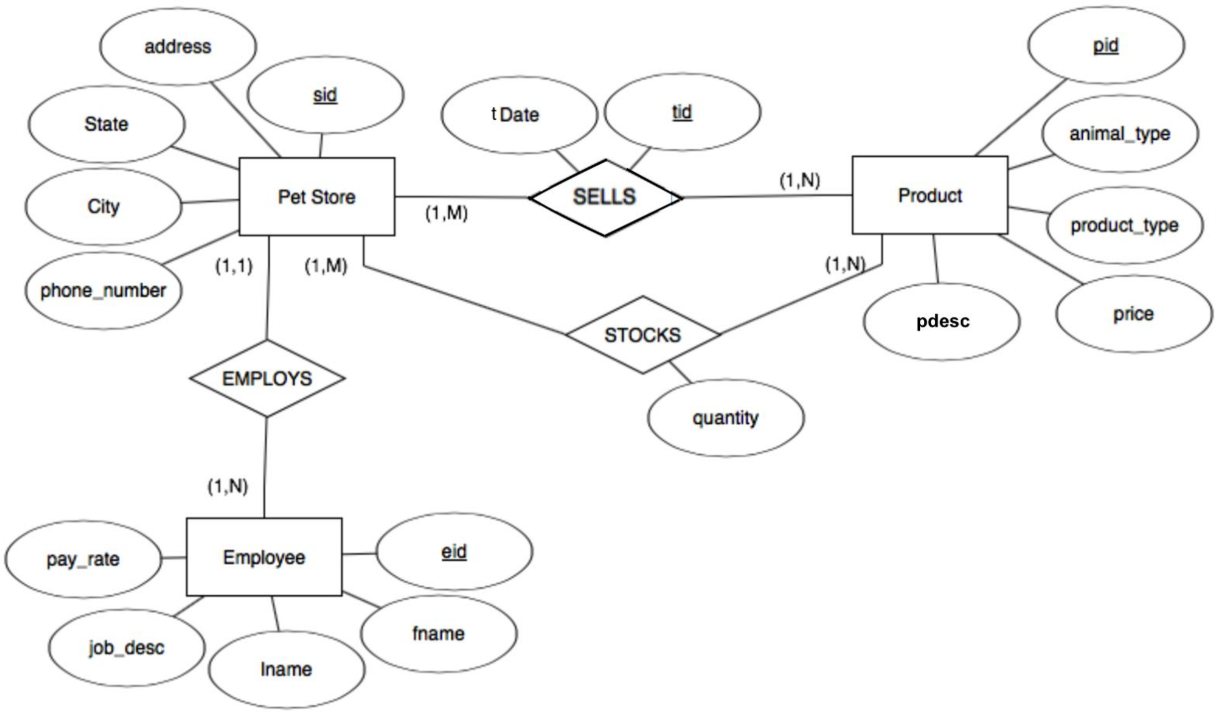
Gathered Requirements

- **Informal**
  Multiple pet stores exist.
  A pet store sells multiple products.
  A pet store stocks multiple products.
  A pet store employs multiple employees.
  An employee is employed by a pet store.
  A product is stocked at multiple pet stores.
  A product is sold at multiple pet stores.

- **Entities**
  A Pet Store Entity with (sid, address, City, State, phone_number) attributes.
  An Employee Entity with (eid, fname, lname, job_desc, pay_rate) attributes.
  A Product Entity with (pid, price,product_type,animal_type,pdesc) attributes.

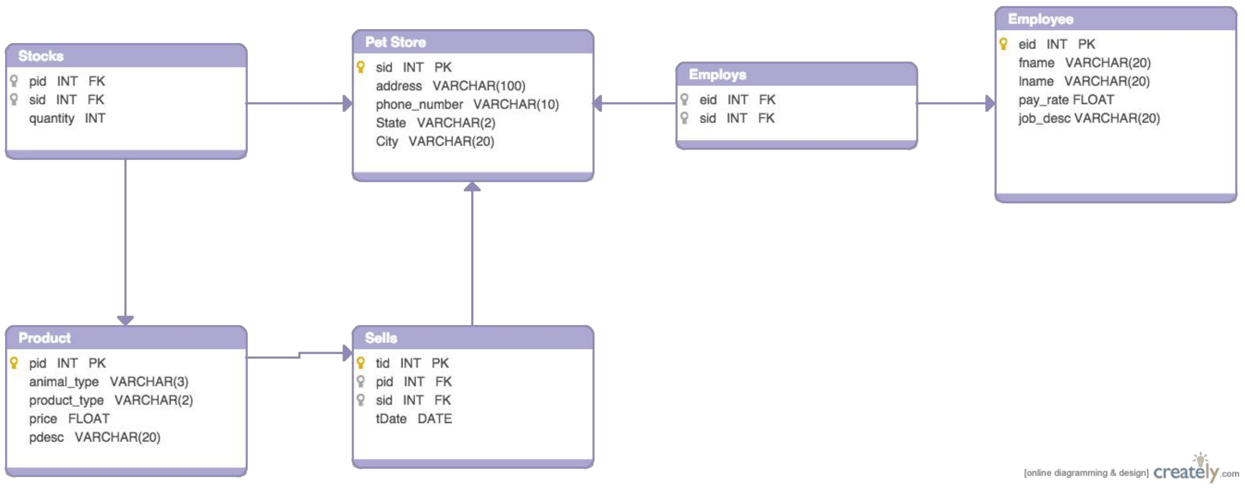
- **Relationships**
  An EMPLOYS relationship with cardinality (1,N).
  Mandatory 1 Pet Store EMPLOYS Mandatory 1 Employee or more.
  A SELLS relationship with attributes (tid, tDate) and cardinality (M,N).
  Mandatory 1 or more Pet Store SELLS Mandatory 1 Product or more.
  A STOCKS relationship with attribute (quantity) and cardinality (M,N).
  Mandatory 1 Pet Store or more STOCKS Mandatory 1 Product or more.

ER Diagram

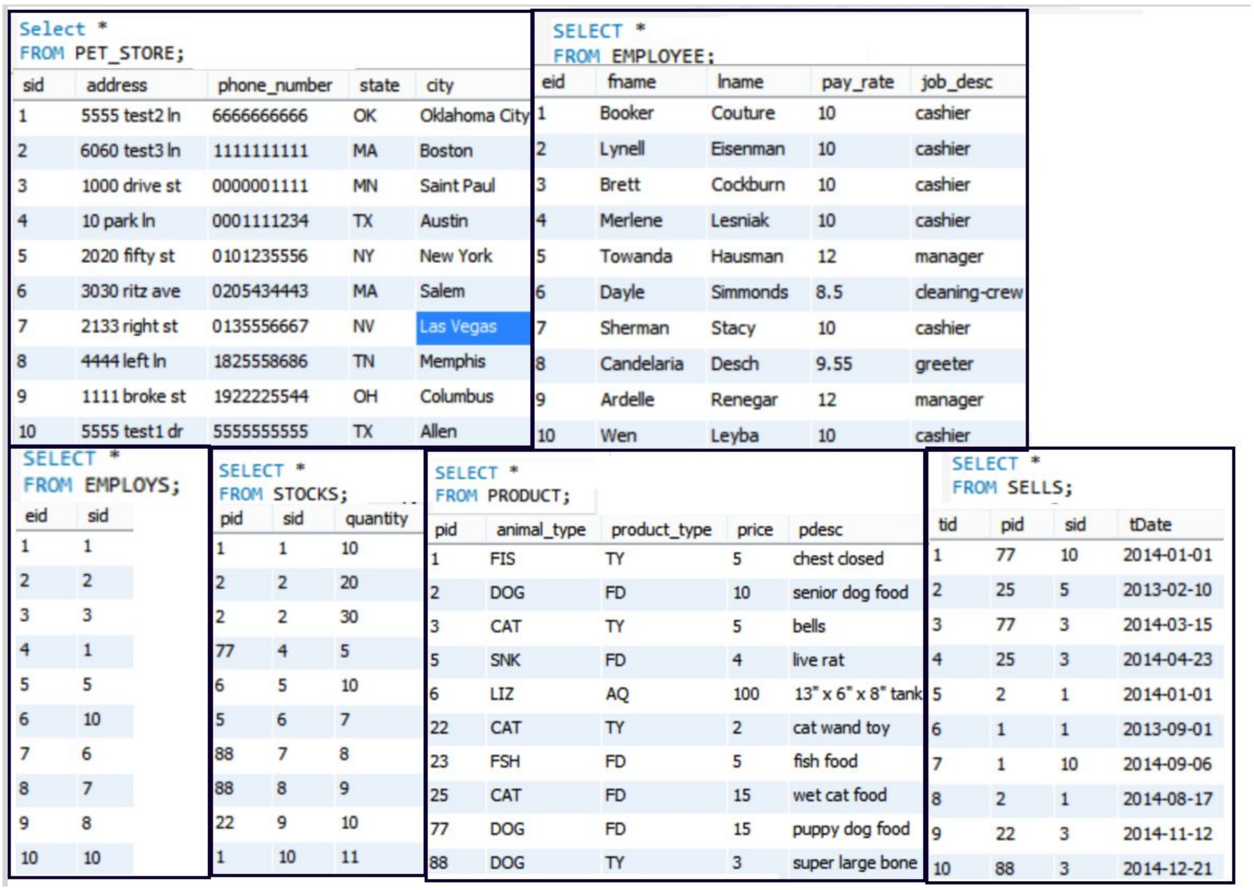


Short Description of ER Diagram
Rectangle - Entity
Circle - Attribute
Parenthesis around attribute name - Composite Attribute
Underline - Unique Attribute
Double Circled - Multi-valued Attribute
(O) - Optional Attribute
Rhombus - Relationship between Entities
Parenthesis - Cardinality

Relational Schema Diagram



Screenshot of Tables



```
Select *
FROM PET_STORE;
```

sid	address	phone_number	state	city
1	5555 test2 ln	6666666666	OK	Oklahoma City
2	6060 test3 ln	1111111111	MA	Boston
3	1000 drive st	0000001111	MN	Saint Paul
4	10 park ln	0001111234	TX	Austin
5	2020 fifty st	0101235556	NY	New York
6	3030 ritz ave	0205434443	MA	Salem
7	2133 right st	0135556667	NV	Las Vegas
8	4444 left ln	1825558686	TN	Memphis
9	1111 broke st	1922225544	OH	Columbus
10	5555 test1 dr	5555555555	TX	Allen

```
SELECT *
FROM EMPLOYEE;
```

eid	fname	lname	pay_rate	job_desc
1	Booker	Couture	10	cashier
2	Lynell	Eisenman	10	cashier
3	Brett	Cockburn	10	cashier
4	Merlene	Lesniak	10	cashier
5	Towanda	Hausman	12	manager
6	Dayle	Simmonds	8.5	cleaning-crew
7	Sherman	Stacy	10	cashier
8	Candelaria	Desch	9.55	greeter
9	Ardelle	Renegar	12	manager
10	Wen	Leyba	10	cashier

```
SELECT *
FROM EMPLOYS;
```

eid	sid
1	1
2	2
3	3
4	1
5	5
6	10
7	6
8	7
9	8
10	10

```
SELECT *
FROM STOCKS;
```

pid	sid	quantity
1	1	10
2	2	20
2	2	30
77	4	5
6	5	10
5	6	7
88	7	8
88	8	9
22	9	10
1	10	11

```
SELECT *
FROM PRODUCT;
```

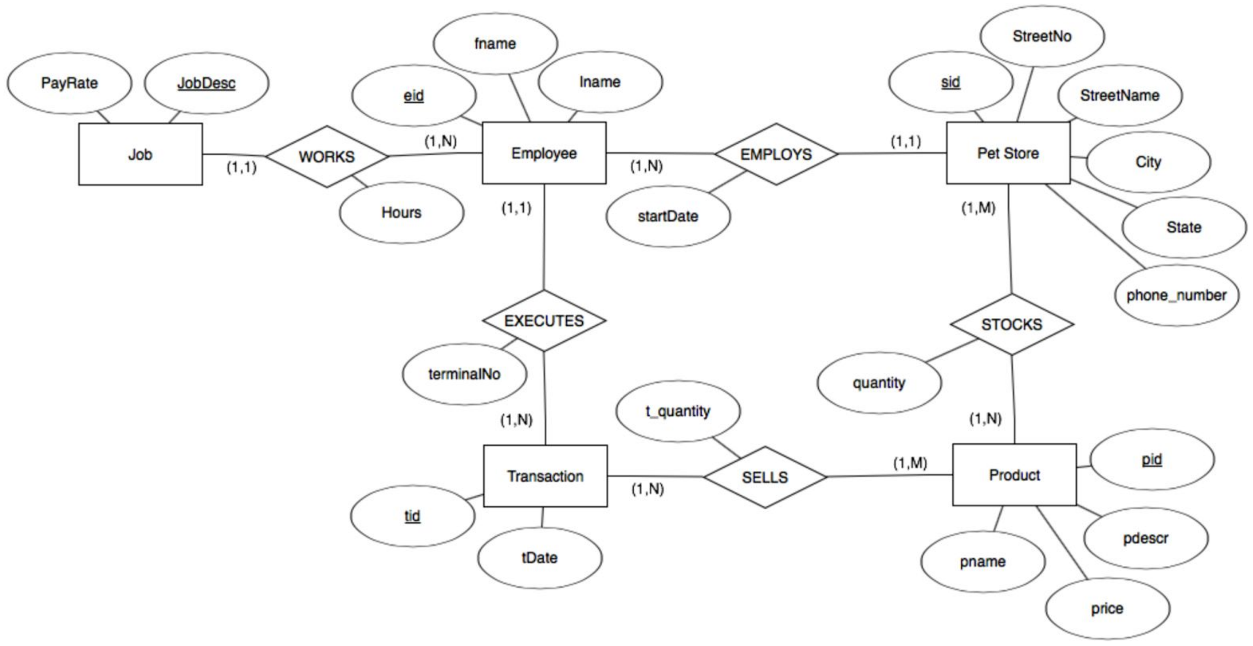
pid	animal_type	product_type	price	pdesc
1	FIS	TY	5	chest closed
2	DOG	FD	10	senior dog food
3	CAT	TY	5	bells
5	SNK	FD	4	live rat
6	LIZ	AQ	100	13" x 6" x 8" tank
22	CAT	TY	2	cat wand toy
23	FSH	FD	5	fish food
25	CAT	FD	15	wet cat food
77	DOG	FD	15	puppy dog food
88	DOG	TY	3	super large bone

```
SELECT *
FROM SELLS;
```

tid	pid	sid	tDate
1	77	10	2014-01-01
2	25	5	2013-02-10
3	77	3	2014-03-15
4	25	3	2014-04-23
5	2	1	2014-01-01
6	1	1	2013-09-01
7	1	10	2014-09-06
8	2	1	2014-08-17
9	22	3	2014-11-12
10	88	3	2014-12-21

Deliverable II

ER Diagram with Normalized Relations



Changes Noted

Pet Store **address was expanded into StreetNo and Street Name in order to follow 1NF.

Employs **Employs has a new attribute startDate in order further describe the relationship between Pet Store and Employee.

Employee **Employee loses PayRate and JobDesc in order to resolve partial dependencies to satisfy 2NF. Without any Transitive Dependencies 3NF is also satisfied.

Job **JobDesc becomes a primary attribute of a Job entity. It is included with PayRate to further describe a Job.

Works **Employee Works a Job. The Hours attribute further describes the relationship.

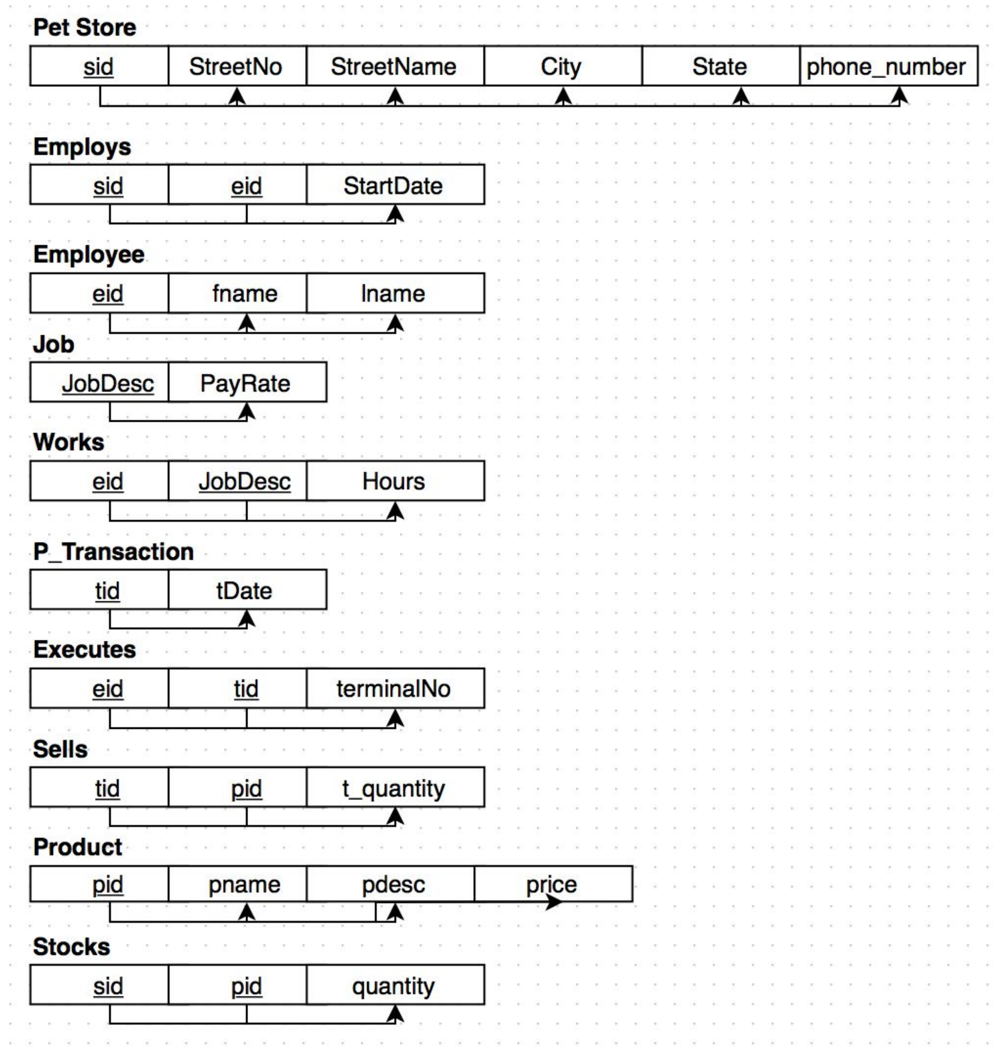
P_Transaction **Transaction is listed as P_Transaction because it ended up being a Key Value for MySQL. The Transaction Sells the Product instead of the Pet Store.

Executes **Employee Executes the Transaction. terminalNo describes where the relationship takes place.

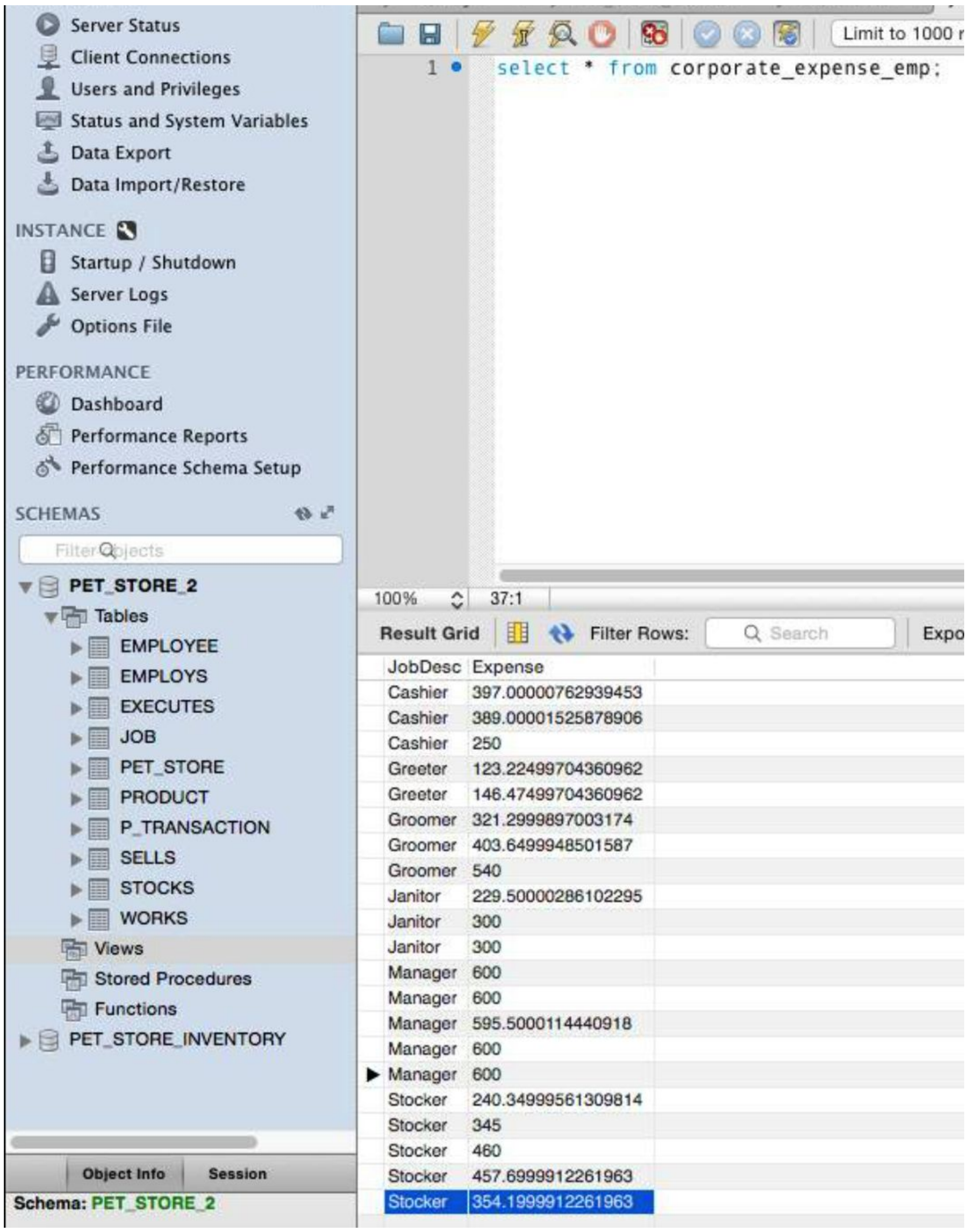
Sells **Transaction Sells Product. t_quantity describes how much Product is sold in a Transaction.

Product **We removed animal_type and product_type because the information seemed unnecessary and added a pname

Dependency Diagram



View



```
select * from corporate_expense_emp;
```

JobDesc	Expense
Cashier	397.00000762939453
Cashier	389.00001525878906
Cashier	250
Greeter	123.22499704360962
Greeter	146.47499704360962
Groomer	321.2999897003174
Groomer	403.6499948501587
Groomer	540
Janitor	229.50000286102295
Janitor	300
Janitor	300
Manager	600
Manager	600
Manager	595.5000114440918
Manager	600
Manager	600
Stocker	240.34999561309814
Stocker	345
Stocker	460
Stocker	457.6999912261963
Stocker	354.1999912261963

corporate_expense_emp takes account of each job description and the total expense on a corporate level since the view lists all the employees of each store. The expense consists of payrate per jobdesc * hours the employee worked.

View Statement

```
CREATE VIEW CORPORATE_EXPENSE_EMP AS

SELECT JOB.JobDesc, (JOB.PayRate * WORKS.Hours) AS Expense

FROM JOB

LEFT JOIN WORKS

ON JOB.JobDesc = WORKS.JobDesc;
```

Python script

The following link is a python script that can handle simple interactions with the database. You can remove/Update/Insert tuples from the command line.

https://github.com/ggreenleaf/PetStoreInterface

To run the script you must have mysql server running on your computer as well as the python mysql connector installed. To run the script you can download the interface.py or use the file that is in the turned in archive. On our computers we had the Database called PET_STORE_2 and the default user is root with the default password. If you are running an instance of the database under a different user these options will need to be changed.