**Project title:** PET WORLD

**Team Members:**

Abhishek Madan (969)

Bhakti Shah (916)

Siddharth Gore (037)

Xiaoyu Zhou (038)

**The choice of database project:**

This project is to help animal-rescue organizations/people to hand over the animals, and to help individuals adopt the pet they need.

**The choice of database engine**:Oracle, MySQL

**DB application technologies**: Oracle SQL Developer, MySQL workbench

**Frameworks**: None

**Languages**: sql, pl\sql

**Final list of functionalities/operations:**

1. A person should be able to register himself (update his details if already exist).
2. The DB user(organization/customer) should be able to see his details
3. A donor (organization/ person) should be able to see the request, whose approval are pending on him.
4. A donor should be able to approve the request (this approval of request should update the columns in the table: request, animal and  adoption history)
5. An adopter, while looking for an animal should be able to

* Look at the breeds offered.
* And then using the breed id know the pets available.

1. A requester should be able to know the adoption history of an animal from the animal id.
2. A requester should be able to browse the medical history of an animal from the animal id.
3. A requester should be able to see the status of the request he made.
4. A donor should be able to put down the information of a new pet he wants to don.

10. A donor should be able to put on re-adoption an animal which he took from the agency

11. Show all the requests made by a person till date.

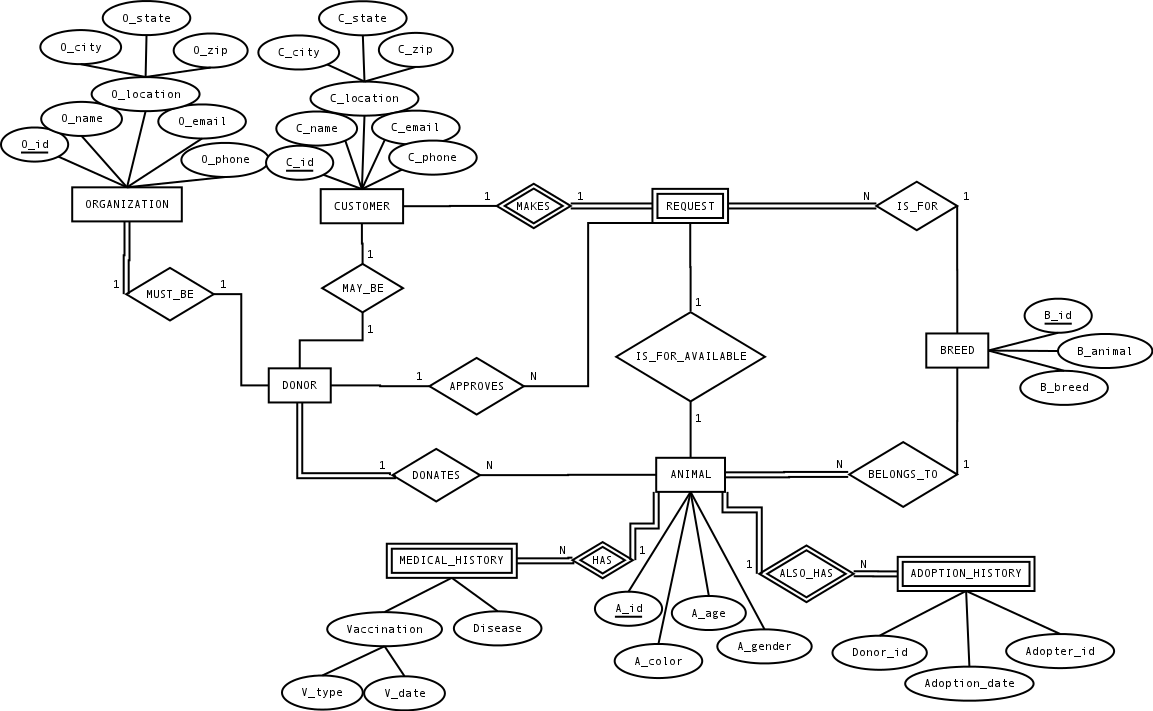
12. A customer can raise a request to adopt a pet.

**Final major areas/components/tasks/Milestones (including completion date) for each team member:**

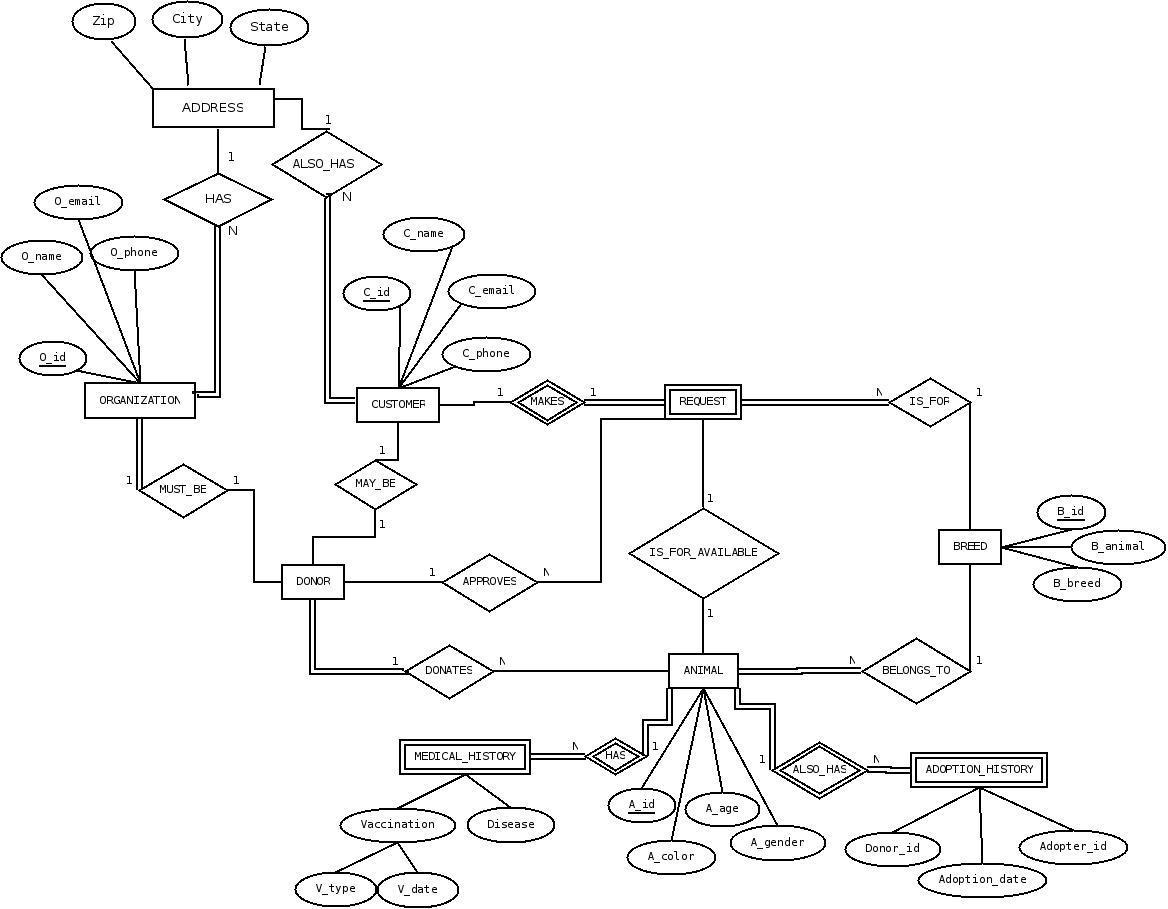
|  |  |  |  |
| --- | --- | --- | --- |
| **FINISH DATE** | **TASK** | **TASK PERFORMER** | **DETAILS** |
| SEP 20 2016 | Project Proposal | ALL TEAM MEMBERS | Group meeting scheduled to draft the proposal |
| OCT 5 2016 | ER Diagram | ALL TEAM MEMBERS | After having individual ER diagram, group meeting scheduled to discuss and create the final version |
| OCT 25 2016 | TABLES: ORGANIZATION,  REQUEST | BHAKTI | Create tables and populated with data |
| OCT 25 2016 | TABLES:  ANIMAL  BREED | XIAOYU | Create tables and populated with data |
| OCT 25 2016 | TABLES:  CUSTOMER  REQUEST | ABHISHEK | Create tables and populated with data |
| OCT 25 2016 | TABLES:  MEDICAL HISTORY  ADOPTION HISTORY | SIDDHARTH | Create tables and populated with data |
| NOV 5 2016 | OPERATIONS | ALL TEAM MEMBERS | All operations were divided among all team members. |
| NOV 7 2016 | OPERATIONS | ALL TEAM MEMBERS | Group meeting conducted to discuss and verify all the queries of the operations. |
| NOV 15 | PROJECT REPORT | ALL TEAM MEMBERS | Divided project report into four sections and each team member worked each section |
| NOV 20 | FINAL ASSEMBLING REPORT | ALL TEAM MEMBERS | Group meeting conducted to discuss final report and create final draft and all pending work. |
| NOV 29 | FINAL CHECK FOR FUNCTIONALITY OF DB AND MISTAKE PROOFING REPORT | ALL TEAM MEMBERS |  |

**ER Diagram:**

OLD VERSION:



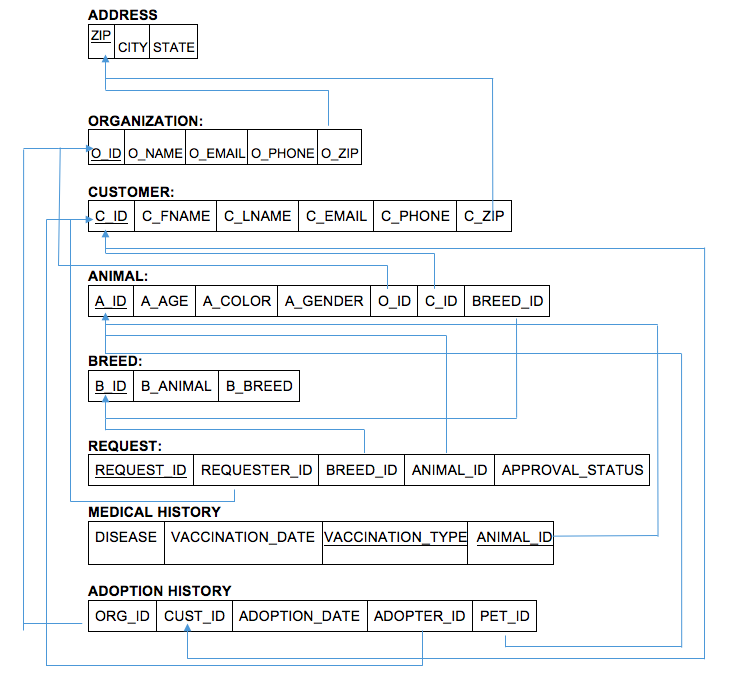
NEW VERSION:



**Any major modifications from the proposal, ERD, EERD and why:**

The older version had Address as an attribute to both organization and customer. In the new version, Address is treated as an entity which has three attributes: zip, City, State. This change makes easier to add new location into database and redundancy. Also, there transitive dependency of attributes which made us think, that it is better idea to create separate entity.

**Final design of database portion:**



**The normal form (3NF, BCNF, etc) of tables/relationship. Justify the reasons if any of them is below 3NF**

ORGANIZATION:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| O\_ID | O\_NAME | O\_EMAIL | O\_PHONE | O\_ZIP |

1NF: YES. O\_ID is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on O\_ID.

3NF: YES. O\_ID directly depends every other non-PK attribute.

CUSTOMER

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| C\_ID | C\_FNAME | C\_LNAME | C\_EMAIL | C\_PHONE | C\_ZIP |

1NF: YES. C\_ID is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on C\_ID.

3NF: YES. C\_ID directly depends every other non-PK attribute.

ADDRESS

|  |  |  |
| --- | --- | --- |
| ZIP | CITY | STATE |

1NF: YES. ZIP is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on ZIP.

3NF: YES. ZIP directly depends every other non-PK attribute.

ANIMAL:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A\_ID | A\_AGE | A\_COLOR | A\_GENDER | O\_ID | C\_ID | Breed\_ID |

1NF: YES. A\_ID is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on A\_ID.

3NF: YES. A\_ID directly depends every other non-PK attribute.

BREED:

|  |  |  |
| --- | --- | --- |
| B\_ID | B\_ANIMAL | B\_BREED |

1NF: YES. B\_ID is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on B\_ID.

3NF: YES. B\_ID directly depends every other non-PK attribute.

REQUEST

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| REQUEST\_ID | REQUESTER\_ID | BREED\_ID | ANIMAL\_ID | APPROVAL\_STATUS |

1NF: YES. REQUEST\_ID is the PK and determines every other non-PK attribute.

2NF: YES. No composite PK, every non-PK attribute fully FDs on REQUEST\_ID.

3NF: YES. REQUEST\_ID directly depends every other non-PK attribute.

MEDICAL\_HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| DISEASE | VACCINATION\_DATE | VACCINATION\_TYPE | ANIMAL\_ID |

1NF: Yes, primary key is present and all other attributes depends on pk.

2NF: No, Reason: further normalization would result into too many small tables.

ADOPTION\_HISTORY (one of ORG\_ID and CUST\_ID is null)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ORG\_ID | CUST\_ID | ADOPTION\_DATE | ADOPTER\_ID | PET\_ID |

1NF: No. No Primary Key present so Not present in 1NF, 2NF or 3NF.

**Specification of each DB object (table, column, view, stored procedure, trigger, etc) and its meaning/purpose:**

**Tables-**

ADDRESS:

|  |  |  |
| --- | --- | --- |
| ZIP | CITY | STATE |

Address table is used to store all the Zip codes, city and state of customer as well as organization. If a new customer or organization is registered with new zip code, then that will be first entered in this table. Zip is unique so, it is used as a primary key. Zip is also used as foreign key in ORGANIZATION as well as CUSTOMER table which reflects the location of that particular ORGANIZATION/CUSTOMER.

ORGANIZATION:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| O\_ID | O\_NAME | O\_EMAIL | O\_PHONE | O\_ZIP |

ORGANIZATION table is used to store information about organization’s name, email, and phone number and zip\_code. O\_zip is the foreign key which is the primary key in the Address Table.

CUSTOMER:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| C\_ID | C\_FNAME | C\_LNAME | C\_EMAIL | C\_PHONE | C\_ZIP |

CUSTOMER table is used to store information about customer’s First name, last name, email Id, phone number and zip\_code. C\_zip is the foreign key which is the primary key in the Address Table.

ANIMAL:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| A\_ID | A\_AGE | A\_COLOR | A\_GENDER | O\_ID | C\_ID | BREED\_ID |

ANIMAL table is used to store information about all the animal listed by Organization who wish to give away the animals from the shelter and Customers who wish to donate their pet’s. O\_id is the foreign key which is the primary key in ORGANIZATION table. The value of O\_ID is null if particular animal is listed by some customer. C\_ID is foreign key which is primary key in CUSTOMER table. The value of the C\_ID can be null if particular animal is listed by some Organization. BREED\_ID is a foreign key which is primary key in the BREED table. Breed\_id is used to determine type of animal and its breed.

BREED:

|  |  |  |
| --- | --- | --- |
| B\_ID | B\_ANIMAL | B\_BREED |

BREED table is to store information of all animal breeds. B\_ANIMAL is animal kind, for example, ‘Cat’, ‘Dog’. B\_BREED is the specific breed of an animal kind, for example, ‘Persian’ for ‘Cat’, 'German Shepherd' for ‘Dog’.

REQUEST:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| REQUEST\_ID | REQUESTER\_ID | BREED\_ID | ANIMAL\_ID | APPROVAL\_STATUS |

REQUEST table is to store the information for every request. REQUESTER\_ID is the Customer ID who made this request, BREED\_ID and ANIMAL\_ID is the information of the animal who is being requested. APPROVAL\_STATUS is the status of this request, it’s ‘N’ means NO, the request is not approved, ‘Y’ means YES, the request is approved, ‘P’ means PENDING, the request is waiting for approved/rejected.

MEDICAL\_HISTORY

|  |  |  |  |
| --- | --- | --- | --- |
| DISEASE | VACCINATION\_DATE | VACCINATION\_TYPE | ANIMAL\_ID |

MEDICAL HISTORY table is used to store the information about different disease conditions that particular animal had or have, vaccination type that is particular animal is given in past and the date of vaccination. Here the primary key is composite primary key because, one particular animal can be vaccinated with more than one vaccine. Animal\_Id is the foreign key, which is the primary key in the ANIMAL table.

ADOPTION\_HISTORY

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ORG\_ID | CUST\_ID | ADOPTION\_DATE | ADOPTER\_ID | PET\_ID |

ADOPTION\_HISTORY table is to store the adoption history of a specific animal whose animal id is PET\_ID. This animal was provided by an organization with ORG\_ID or a customer with CUST\_ID and was adopted by a customer with ADOPTER\_ID. ADOPTION\_DATE is the date of the adopter adopted this animal.

**Procedures-**

1. **approve\_pet\_request -** Procedure to approve the request for a pet by changing the approval status of the request to ‘Y’ in the request table.
2. **enter\_pet\_medical\_history-** Procedure to enter the medical history of the pet. The records can be added as and when the pet is vaccinated.
3. **register\_new\_customer-** Procedure to register a new customer.
4. **register\_new\_organization-** Procedure to register a new organization.
5. **register\_new\_pet-** Procedure to register a new pet.
6. **request\_a\_pet-** Procedure to raise a request for a pet. If the pet is not available, raise the request as a wish list from the customer.
7. **show\_animal\_adoption\_history-** Show the adoption history of an animal.
8. **show\_animal\_for\_a\_breed-** Show all the animals available for the breed.
9. **show\_approval\_pending\_on\_me-** Shows to the pet owner, the approvals pending for a request made by a adopter.

**10.show\_available\_breeds-** Lists all the available breeds with the Agency.

**11.show\_details\_of\_customer-** Show all the details of a customer/organization from its id.

**12.show\_my\_request\_info-** Gets the status of the request raised by an adopter.

**13.show\_animal\_medical\_history**- Shows the medical history for a given pet.

**14.register\_a\_wishlist-** This procedure allows the customer to register a request for a pet of a particular breed even when it is now available with the Agency.

**Any specific functionality involving modification of more than one table:**

Eg 1: When a request for a particular animal is approved, then there are three table that get affected by such transaction. In the animal table, updates are made on C\_ID which reflects the owner of that particular animal. In the Request table, approval status changes. In Adoption History table, there will be an update on Adopter’s Id and Adoption Date.

Eg 2: When any new customer or organization are willing to be a part of pet’s World and if the zip code of that particular organization is not present in the database then in order to maintain the referential integrity, we first need to add the zip code in address table and then it will be added to customer/organization table.

**Any implicit/explicit database transactions initiated from DB server side:**

Implicit: every query statement is implicit transaction;

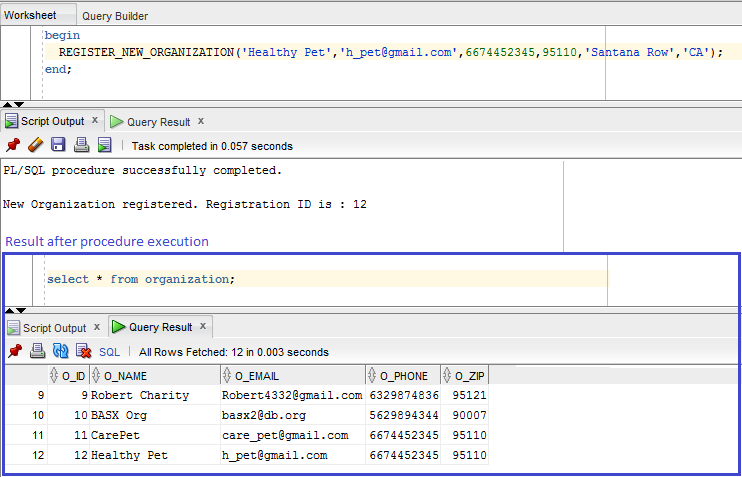
Explicit: none.

**Any additional DB objects/concepts utilized (view, stored procedure, trigger, index, isolation, CC, etc.):**

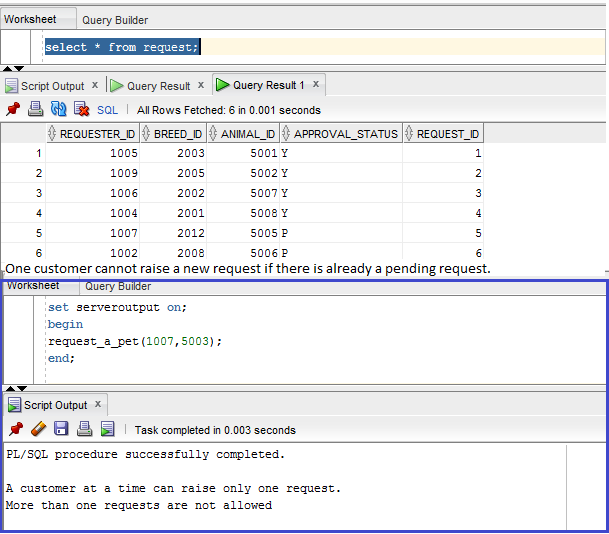
Stored Procedures were implemented in order to make presentation time efficient.

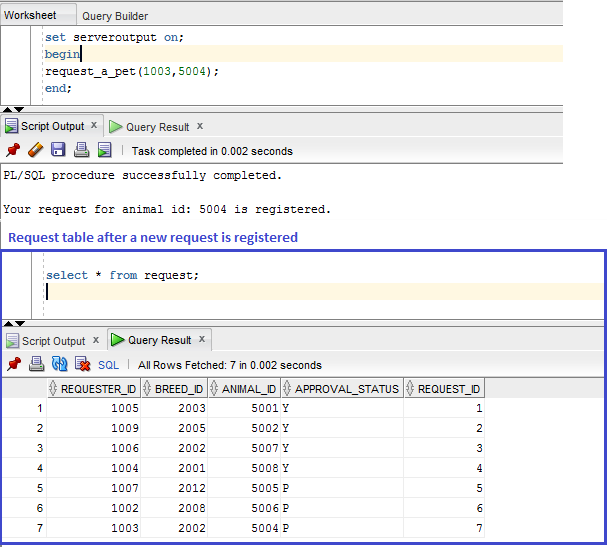
**Sample execution of query/sp (screen shots, etc.) for important operations:**

1. Registering a new Organization.



2. A customer can make only one request.



3. Register a request for a pet.

**Source code:**

see separate files

**Any unique designs you are proud of:**

stored procedures.

**Test plan execution:**

In order to verify the results, we verified the state of the database before and after the execution of procedures. Our sample size was 5.

**Any potential improvements:**

A user-friendly GUI.

**Project postmortem**:

When we mapped our ERD to tables and did the normalization (specifically, 3NF), there was some redundancy because both the ORGANIZATION and CUSTOMER have LOCATION attribute, we improved it by moving location into a separate table then it is easier to add new / delete existing addresses.