

Animal Shelter Database System Project

Project Objective:

Organizing tons of data in every organization is important to effectively manage operations and streamline processes. QUELIFE is an animal shelter organization for which we will be designing a database system. This organization has multiple branches spread across Wisconsin, Illinois, Missouri, Montana, California, Texas, and Nevada. This database will encompass primary details related to facilities, animal descriptions and transfer details, sponsor details and employee details. We are also ensuring to maintain high quality of the database using normalization.

Requirement Analysis:

We are analysing the requirements to setup a database for this organization. We have multiple entities to consider along with their relationships. Here we have multiple branches of shelter at multiple locations with various facilities. Animal related records including animal information, animal features, adoption records and surrender details are need as well as the history of each animal. We are enabling the option to transfer of animals from one branch to another. Employee details, sponsor details and payment details are required to track the working of the shelter. Applying normalization to the tables and splitting the tables to ensure that there is no information loss.

Assumptions:

1. Animals can be transferred from one branch to another
2. Each shelter branch will have multiple employees
3. Each shelter branch will have a branch manager
4. Employees work only in one shelter branch only (no transfer of employees)
5. An animal can be adopted/surrendered to the shelter for various reasons multiple times
6. A vaccination is given each time an animal is detected with any disease
7. Sponsors can support multiple shelter branches
8. Full payment must be done in a single payment for adoption of an animal
9. Each medical condition of an animal is treated by only one veterinarian
10. Veterinarians can treat multiple conditions of multiple animals
11. Each transfer of an animal involves transfer from one branch to another
12. Each veterinary doctor can have multiple specializations
13. All contacts should not contain spaces
14. Employees work on hourly basis and hourly pay varies from employee to employee
15. Any animal is considered as ready for adoption if its vaccination status is up to date
16. Gross pay is the total pay for an employee without any deduction
17. Net pay is the pay of an employee after tax deduction
18. Employee salary is an estimate considering the maximum number of hours they can work
19. Discount is applied on adoption fee if the amount is greater than 750 by 5% and when 1000 by 10%
20. Surrender reason cannot exceed 20 characters
21. Date of birth of an animal cannot be greater than date of acquisition
22. Employee department is selected depending on their education only
23. Employee has a unique hourly pay until unless the pay is changed and the table is updated
24. Each breed determines the type of animal
25. Address of the shelter has an unique pincode

Updated Entities and Relationships:

1. **shelter_branch:**
 - Entity Description: This entity contains details about the geographic location and details of each branch
 - Attributes: shelter_name, shelter_loc, address, shelter_contact
 - Relationships:
 - One-to-one relationship with facilities
 - Many-to-many relationship with animal_info
 - One-to-many relationship with employees_staff_details
 - Many-to-many relationship with animal_transfer_records
 - Many-to-many relationship with sponsor
 - Many-to-one relationship with shelter_loc_details
2. **shelter_loc_details (new table)**
 - Entity Description: This entity contains details about the shelter location of shelter
 - Attributes: shelter_loc, pin_code
 - Relationships:
 - One-to-many relationship with shelter_branch
3. **facilities:**
 - Entity Description: This entity represents amenities operated by each branch.
 - Attributes: fid, f_capacity, f_ani_count, f_area, play_areas, temp_control, outdoor_space
 - Relationships:
 - One-to-one relationship with shelter_branch
4. **animal_info:**
 - Entity Description: This entity contains information about the animal.
 - Attributes: animal_id, a_breed, a_dob, date_of_acquisition, a_vaccine_status, animal_features_id
 - Relationships:
 - Many-to-many relationship with shelter_branch (as we have transfer facility)
 - One-to-one relationship with animal_features
 - Many-to-many relationship with surrender_details
 - Many-to-many relationship with adoption_details
 - One-to-many relationship with animal_transfer_records
 - Many-to-many relationship with nutrition
 - Ternary relationship with history of the animal which includes: grooming history, training history and medical history
 - Many-to-one relationship with *animal_breed_info*
5. **animal_breed_info (new table)**
 - Entity Description: This entity contains information about the animal breed.
 - Attributes: a_breed, a_type
 - Relationships:
 - One-to-many relationship with animal_info
6. **animal_features:**
 - Entity Description: This entity contains physical attributes of the animal.
 - Attributes: animal_id, a_color, a_weight, a_height, a_eye_color, a_condition

- Relationships:
 - One-to-one relationship with animal_info

7. adoption_details:

- Entity Description: This entity contains information about the adoption of an animal
- Attributes: ad_ID, animal_id, ad_name, ad_contact, ad_date, ad_fee.
- Relationships:
 - Many-to-many relationship with animal_info
 - One-to-one relationship with payment_details

8. surrender_details:

- Entity Description: This entity contains details about returning of the animal to the shelter
- Attributes: s_id, s_date, s_name (person who brings animals), s_reason.
- Relationships:
 - Many-to-many relationship with animal_info

9. nutrition:

- Entity Description: This contains information about diet and nutrition of each animal
- Attributes: food_id, food_type, food_quantity
- Relationships:
 - Many-to-many relationship with animal_info

10. training_history:

- Entity Description: This entity contains details about training the animals
- Attributes: training_ID, trainer_name, training_date, training_type, training_stage.
- Relationships:
 - Many-to-one relationship with animal_info
 - Many-to-many relationship with employees_staff_details

11. grooming_history:

- Entity Description: This entity contains details about grooming the animals
- Attributes: grooming_id, groomer_name, grooming_date, grooming_type.
- Relationships:
 - Many-to-one relationship with animal_info
 - Many-to-many relationship with employees_staff_details

12. medical_history:

- Entity Description: This entity contains details about the medical history of each animal
- Attributes: medical_id, disease, treatment_date
- Relationships:
 - Many-to-one relationship with animal_info
 - Many-to-many relationship with vet_id
 - One-to-one relationship between medical_history and vaccination
 - Many-to-one relationship with disease_mitigation

13. disease_mitigation (*new table*)

- Entity Description: This entity contains details about the disease_mitigation of each animal.
- Attributes: disease, vaccination_id
- Relationships:
 - One-to-many relationship with medical_history
 - One-to-one relationship between disease_mitigation and vaccination

14. vaccination:

- Entity Description: This entity contains details about the vaccination of each animal
- Attributes: vaccination_id, vaccination_name, vaccination_date, vaccination_dosage
- Relationships:
 - One-to-one relationship between medical_history and vaccination

15. employees_staff_details:

- Entity Description: This entity contains details about each worker or employee at the animal shelter QUELIFE
- Attributes: emp_id, emp_name, emp_contact, emp_training, emp_education, emp_criminal_record
- Relationships:
 - Many-to-one relationship with shelter_branch
 - Many-to-many relationship with grooming_history
 - Many-to-many relationship with training_history
 - Unary relationship – manages – branch manager
 - Many-to-one relationship with employees_stage

16. employees_stage (*new table*)

- Entity Description: This entity contains details about each employee stage at the animal shelter QUELIFE
- Attributes: emp_level , emp_department, emp_education
- Relationships:
 - One-to-many relationship with employees_staff_details

17. sponsor:

- Entity Description: This entity contains details about sponsors given to shelter_branch
- Attributes: sponsor_id, sponsor_name, sponsor_contact, sponsorship_type.
- Relationships:
 - Many-to-many relationship with shelter_branch
 - One-to-many relationship with payment_details

18. payment_details:

- Attributes: payment_ID, amount, payment_Date, payment_type.
- Relationships:
 - One-to-one relationship with adoption_details
 - Many-to-one relationship with sponsor

19. veterinary:

- Entity Description: This entity contains details of the veterinarian
- Attributes: vet_id, vet_name, vet_contact, education
- Relationships:
 - Many-to-many relationship with medical_history
 - Many-to-one relationship with vet_edu

20. *Vet_edu(new table):*

- Entity Description: This entity contains details of the veterinarian education
- Attributes: vet_name, vet_contact, work_exp
- Relationships:
 - One-to-many relationship with veterinary

21. *animal_transfer_records:*

- Entity Description: This entity contains details of transfer of animals from one branch to another
- Attributes: transfer_id, transfer_date, transfer_from, transfer_to.
- Relationships:
 - Many-to-one relationship with animal_info
 - Many-to-many relationship with shelter_branch

22. *employee_payroll:*

- Entity Description: This entity contains details of the working hours of each employee.
- Attributes: swipe_id, swipe_date, check_in_time , check_out_time, emp_id
- Relationships:
 - Many-to-one relationship with employees_staff_details
 - Many-to-one relationship with employee_pay

23. *employee_pay(new table):*

- Entity Description: This entity contains details of the hourly pay of each employee.
- Attributes: emp_id, hourly_pay
- Relationships:
 - One-to-many relationship with employee_pay
 - Many-to-one relationship with employees_staff_details

UPDATED SCHEMA: (Primary Key: Underlined, Foreign Key: Italics)

shelter_branch (shelter_id, shelter_name, shelter_loc, address, shelter_contact)

animal_info (animal_id, a_breed, a_dob, date_of_acquisition, a_vaccine_status, animal_features_id)

animal_features (animal_features_id, a_color, a_weight, a_height, a_eye_color, a_condition)

animal_transfer_records (transfer_id, transfer_date, transfer_from, transfer_to, animal_id)

grooming_history (grooming_id, groomer_name, grooming_date, grooming_type)

nutrition (food_id, food_type, food_quantity)

sponsor (sponsor_id, sponsor_name, sponsor_contact, sponsorship_type)

surrender_details (s_id, s_date, s_name, s_reason)

training_history (training_ID, trainer_name, training_date, training_type, training_stage)

veterinary (vet_id, vet_name, vet_contact, education)

vet_specialization (vet_id, specialization)

medical_history (medical_id, disease, treatment_date)

payment_details (payment_ID, sponsor_id, amount, payment_Date, payment_type)

employees_staff_details (emp_id, emp_name, emp_contact, emp_training, emp_education, emp_criminal_record)

facilities (facilities_id, shelter_id, f_capacity, f_ani_count, f_area, play_areas, temp_control, outdoor_space)

adoption_details (ad_ID, payment_ID, ad_name, ad_contact, ad_date, ad_fee)

vaccination (vaccination_id, vaccination_name, vaccination_date, vaccination_dosage)

employee_payroll (swipe_id, emp_id, swipe_date, check_in_time, check_out_time)

perform (grooming_id, emp_id)

examines (vet_id, medical_id)

support (shelter_id, sponsor_id)

contain (animal_id, ad_ID)

need (food_id, animal_id)

submit (s_id, animal_id)

accommodate (shelter_id, animal_id)

transfers (shelter_id, transfer_id)

done_by (training_ID, emp_id)

stores (animal_id, training_history_id, grooming_history_id, medical_history)

shelter_loc_details (shelter_loc, pin_code)
animal_breed_info (a_breed, a_type)
disease_mitigation (disease, vaccination_id)
employees_stage(emp_level, emp_department, emp_education)
vet_edu (vet_name, vet_contact, work_exp)
employee_pay (emp_id, hourly_pay)

Database Normalization (Part 6):

Database normalization is necessary to design high quality tables. This step is necessary to maintain redundancy of the tables in such a way that no dependant information is lost due to deletion of any data rows. To normalize the tables in the database it is necessary to identify the functional dependencies in each table. Here is a list of functional dependencies we have identified in each table:

1. **shelter_branch:**
shelter_id \rightarrow shelter_name, shelter_loc (shelter_id uniquely determines shelter_name and shelter_loc)

shelter_id, shelter_name \rightarrow address, shelter_contact (For each tuple of shelter_ID and shelter_name, there is a unique address and contact)

shelter_loc \rightarrow pin_code (shelter_loc uniquely determines pin_code)
2. **facilities:**
facilities_id \rightarrow f_capacity, f_ani_count, temp_control, outdoor_space, shelter_id (shelter_id uniquely determines shelter_name and shelter_loc)

facilities_id, outdoor_space \rightarrow f_area, play_areas
3. **animal_info:**
animal_id \rightarrow a_type, a_breed, a_dob, date_of_acquisition, a_vaccine_status
animal_id, a_type \rightarrow animal_features_id
a_breed \rightarrow a_type
4. **animal_features:**
animal_features_id \rightarrow a_color, a_weight, a_height, a_condition, a_eye_color
animal_features_id, a_condition \rightarrow a_color, a_weight, a_height
5. **adoption_details:**
ad_ID \rightarrow ad_name, ad_contact, ad_date, ad_fee, payment_ID
ad_ID, ad_date \rightarrow payment_ID, ad_name, ad_contact
6. **surrender_details:**
s_id \rightarrow s_date, s_name

s_id, s_date -> s_reason

7. **nutrition:**

food_id -> food_type
food_id, food_type -> food_quantity

8. **training_history:**

training_ID -> trainer_name, training_date, training_type
training_ID, training_date -> training_stage

9. **grooming_history:**

grooming_id -> groomer_name, grooming_date
grooming_id, groomer_name -> grooming_type

10. **medical_history:**

medical_id -> disease, vaccination_id
medical_id, disease -> vaccination_id, treatment_date

11. **vaccination:**

vaccination_id -> vaccination_name, vaccination_date
vaccination_id, vaccination_name -> vaccination_dosage

12. **employees_staff_details:**

emp_id -> emp_name, emp_contact, emp_training, emp_education, emp_criminal_record
emp_id, emp_contact -> emp_criminal_record
emp_education -> emp_level
emp_education -> emp_department

13. **sponsor:**

sponsor_id -> sponsor_name, sponsor_contact
sponsor_id, sponsor_name, sponsor_contact -> sponsorship_type

14. **payment_details:**

payment_ID -> sponsor_id, payment_Date, payment_type
payment_ID, sponsor_id -> amount, payment_Date, payment_type

15. **veterinary:**

vet_id -> vet_name, vet_contact, education, work_exp
vet_name, vet_contact -> work_exp

16. **animal_transfer_records:**

transfer_id -> transfer_date, animal_id
transfer_id, animal_id -> transfer_date, transfer_from, transfer_to

17. **employee_payroll:**

swipe_id -> swipe_date, check_in_time, check_out_time, emp_id
emp_id -> hourly_pay

Considering the FDs mentioned above, we are now checking if the tables are normalized. To normalize database system into BCNF preserving dependencies and maintaining lossless join, we are analysing each table and then decomposing into new tables wherever necessary. We can say that a schema or table is in BCNF if all the functional dependencies for that table

1. **shelter_branch:**

$shelter_id \rightarrow shelter_name, shelter_loc$:

Proof:

This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $shelter_id$) is the primary key which implies it is a super key.

$shelter_id^+ = \{shelter_id, shelter_name, shelter_loc, pincode, shelter_contact\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$shelter_id, shelter_name \rightarrow address, shelter_contact$:

Proof:

This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $shelter_id, shelter_name$) is a super key as we can determine all other attributes using these 2.

Proof:

$(shelter_id, shelter_name)^+ = \{shelter_id, shelter_name, shelter_loc, pincode, shelter_contact\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$shelter_loc \rightarrow pin_code$: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: $shelter_loc$) is not a super key and therefore there is a need of a new table. Shelter location cannot be used to determine each row in the table.

So, we are splitting this table into 2:

shelter_branch: (old table)

$shelter_id$
 $shelter_name$
 $shelter_loc$
 $address$
 $shelter_contact$

shelter_loc_details (new table)

$shelter_loc$
 pin_code

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created

$shelter_id \rightarrow shelter_name, shelter_loc$: This dependency is preserved in the old table

$shelter_id, shelter_name \rightarrow address, shelter_contact$: This dependency is preserved in the old table

$shelter_loc \rightarrow pin_code$: This dependency is preserved in the new table

Thus both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

$R1 \cap R2 \rightarrow R1 - R2$ or $R1 \cap R2 \rightarrow R2 - R1$ (where R1 is the old table and R2 is the new table)
 $R1 \cap R2 = \{ \text{shelter_loc} \}$

$R1 \cap R2 \rightarrow R1 - R2 = \text{shelter_loc} \rightarrow \{\text{shelter_id}, \text{shelter_name}, \text{shelter_loc}, \text{address}, \text{shelter_contact}\}$

$R1 \cap R2 \rightarrow R2 - R1 = \text{shelter_loc} \rightarrow \text{pin_code}$

Second condition is satisfied and thus this join is said to be lossless.

2. **facilities:**

$\text{facilities_id} \rightarrow f_capacity, f_ani_count, temp_control, outdoor_space, shelter_id$:

Proof:

This functional dependency satisfies the requirements of BCNF as the determinant (LHS: facilities_id) is the primary key which implies it is a super key.

$\text{facilities_id} + = \{ \text{facilities_id}, \text{shelter_id}, f_capacity, f_ani_count, f_area, play_areas, temp_control, outdoor_space \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{facilities_id}, \text{outdoor_space} \rightarrow f_area, play_areas$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: facilities_id, outdoor_space) is a super key as we can determine all other attributes using these 2.

$(\text{facilities_id}, \text{outdoor_space}) + = \{ \text{facilities_id}, \text{shelter_id}, f_capacity, f_ani_count, f_area, play_areas, temp_control, outdoor_space \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

This table satisfies conditions of BCNF.

3. **animal_info:**

$\text{animal_id} \rightarrow a_type, a_breed, a_dob, date_of_acquisition, a_vaccine_status$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: animal_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{animal_id}, a_type \rightarrow animal_features_id$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: animal_id, a_type) is a super key as we can determine all other attributes using these 2.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$a_breed \rightarrow a_type$: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: a_breed) is not a super key and therefore there is a need of a new table. Breed of animal cannot be used to determine other attributes in the table.

So, we are splitting this table into 2:

animal_info: (old table)

animal_id

```

a_breed
a_dob
date_of_acquisition
a_vaccine_status
animal_features_id
animal_breed_info (new table)
a_breed
a_type

```

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created

$\text{animal_id} \rightarrow \text{a_type}, \text{a_breed}, \text{a_dob}, \text{date_of_acquisition}, \text{a_vaccine_status}$: This dependency is preserved in the old table

$\text{animal_id}, \text{a_type} \rightarrow \text{animal_features_id}$: This dependency is preserved in the old table
 $\text{a_breed} \rightarrow \text{a_type}$: This dependency is preserved in the new table

Thus, both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

$\text{R1} \cap \text{R2} \rightarrow \text{R1} - \text{R2}$ or $\text{R1} \cap \text{R2} \rightarrow \text{R2} - \text{R1}$ (where R1 is the old table and R2 is the new table)
 $\text{R1} \cap \text{R2} = \{\text{a_breed}\}$

$\text{R1} \cap \text{R2} \rightarrow \text{R1} - \text{R2} = \text{a_breed} \rightarrow \{\text{animal_id}, \text{a_dob}, \text{date_of_acquisition}, \text{a_vaccine_status}, \text{animal_features_id}\}$

$\text{R1} \cap \text{R2} \rightarrow \text{R2} - \text{R1} = \text{a_breed} \rightarrow \text{a_type}$

Second condition is satisfied and thus this join is said to be lossless.

4. ***animal_features:***

$\text{animal_features_id} \rightarrow \text{a_color}, \text{a_weight}, \text{a_height}, \text{a_condition}, \text{a_eye_color}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $\text{animal_features_id}$) is primary key which implies it is a super key.

$\{\text{animal_features_id}^+\} = \{\text{animal_features_id}, \text{a_color}, \text{a_weight}, \text{a_height}, \text{a_eye_color}, \text{a_condition}\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{animal_features_id}, \text{a_condition} \rightarrow \text{a_color}, \text{a_weight}, \text{a_height}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $\text{animal_features_id}, \text{a_condition}$) is a super key as animal_feature_id and a_condition can be used to get the other details in other attributes.

$(\text{animal_features_id}, \text{a_condition})^+ = \{\text{animal_features_id}, \text{a_condition}, \text{a_color}, \text{a_weight}, \text{a_height}, \text{a_eye_color}\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

5. ***adoption_details:***

$ad_ID \rightarrow ad_name, ad_contact, ad_date, ad_fee, payment_ID$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: ad_ID) is the primary key which implies it is a super key.

$ad_ID+ = \{ ad_ID, payment_ID, ad_name, ad_contact, ad_date, ad_fee \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$ad_ID, ad_date \rightarrow payment_ID, ad_name, ad_contact$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: ad_ID, ad_date) is a super key as ad_ID and ad_date can be used to get the other details in other attributes.

$ad_ID, ad_date+ = \{ ad_ID, ad_date, payment_ID, ad_name, ad_contact, ad_fee \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

6. **surrender_details:**

$s_id \rightarrow s_date, s_name$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: s_id) is the primary key which implies it is a super key.

$s_id+ = \{ s_id, s_date, s_name, s_reason \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$s_id, s_date \rightarrow s_reason$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: s_id, s_date) is a super key as s_id and s_date can be used to get the other details in other attributes.

$s_id, s_date+ = \{ s_id, s_date, s_name, s_reason \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

7. **nutrition:**

$food_id \rightarrow food_type$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: food_id) is the primary key which implies it is a super key.

$food_id+ = \{ food_id, food_type, food_quantity \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$food_id, food_type \rightarrow food_quantity$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: food_id, food_type) is a super key as food_id and food_type can be used to get the other details in other attributes.

$(food_id, food_type)+ = \{ food_id, food_type, food_quantity \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

8. **training_history:**

$training_ID \rightarrow trainer_name, training_date, training_type$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: training_ID) is the primary key which implies it is a super key.

$\text{training_ID}+ = \{\text{training_ID}, \text{trainer_name}, \text{training_date}, \text{training_type}, \text{training_stage}\}$
As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{training_ID}, \text{training_date} \rightarrow \text{training_stage}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $\text{training_ID}, \text{training_date}$) is a super key as training_ID and training_date can be used to get the other details in other attributes.
 $\text{training_ID}, \text{training_date}+ = \{\text{training_ID}, \text{training_date}, \text{trainer_name}, \text{training_type}, \text{training_stage}\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

9. **grooming_history:**

$\text{grooming_id} \rightarrow \text{groomer_name}, \text{grooming_date}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: grooming_id) is the primary key which implies it is a super key.

$\text{grooming_id}+ = \{\text{grooming_id}, \text{groomer_name}, \text{grooming_date}, \text{grooming_type}\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{grooming_id}, \text{groomer_name} \rightarrow \text{grooming_type}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $\text{grooming_id}, \text{groomer_name}$) is a super key as grooming_id and groomer_name can be used to get the other details in other attributes.
 $\text{grooming_id}, \text{groomer_name}+ = \{\text{grooming_id}, \text{groomer_name}, \text{grooming_date}, \text{grooming_type}\}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

10. **medical_history:**

$\text{medical_id} \rightarrow \text{disease}, \text{vaccination_id}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: medical_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{medical_id}, \text{disease} \rightarrow \text{vaccination_id}, \text{treatment_date}$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: $\text{medical_id}, \text{disease}$) is a super key as we can determine all other attributes using these on LHS.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$\text{disease} \rightarrow \text{vaccination_id}$: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: disease) is not a super key and therefore there is a need of a new table. Disease name cannot be used to determine each row in the table.

So, we are splitting this table into 2:

$\text{medical_history: (old table)}$

medical_id

disease

treatment_date
disease_mitigation (new table)
disease
vaccination_id

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created
medical_id -> disease, vaccination_id: : This dependency is preserved in the old table
medical_id, disease -> vaccination_id, treatment_date: : This dependency is preserved in the old table
disease -> vaccination_id: : This dependency is preserved in the new table

Thus, both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

$R1 \cap R2 \rightarrow R1 - R2$ or $R1 \cap R2 \rightarrow R2 - R1$ (where R1 is the old table and R2 is the new table)
 $R1 \cap R2 = \{ a_breed \}$

$R1 \cap R2 \rightarrow R1 - R2 = disease \rightarrow \{ medical_id, treatment_date \}$
 $R1 \cap R2 \rightarrow R2 - R1 = disease \rightarrow vaccination_id$

Second condition is satisfied and thus this join is said to be lossless.

11. vaccination:

vaccination_id -> vaccination_name, vaccination_date: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: vaccination_id) is the primary key which implies it is a super key.

$vaccination_id^+ = \{ vaccination_id, vaccination_name, vaccination_date, vaccination_dosage \}$
As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

vaccination_id, vaccination_name -> vaccination_dosage: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: vaccination_id, vaccination_name) is a super key as vaccination_id and vaccination_name can be used to get the other details in other attributes.

$vaccination_id, vaccination_name + = \{ vaccination_id, vaccination_name, vaccination_date, vaccination_dosage \}$

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

12. employees_staff_details:

emp_id -> emp_name, emp_contact, emp_training, emp_education, emp_crimal_record:
This functional dependency satisfies the requirements of BCNF as the determinant (LHS: emp_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

emp_id, emp_contact -> emp_criminal_record: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: emp_id, emp_contact) is a super key as we can determine all other attributes using these on LHS.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

emp_education -> emp_level: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: emp_education) is not a super key and therefore there is a need of a new table. emp_education name cannot be used to determine each row in the table.

emp_education -> emp_department: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: emp_education) is not a super key and therefore there is a need of a new table. emp_education name cannot be used to determine each row in the table.

So, we are splitting this table into 2:

employees_staff_details: (old table)

emp_id
emp_name
emp_contact
emp_training
emp_education
emp_crimeal_record

employees_stage (new table)

emp_education
emp_department
emp_level

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created

$\text{emp_id} \rightarrow \text{emp_name}$, emp_contact , emp_training , emp_education , $\text{emp_criminal_record}$:

This dependency is preserved in the old table

$\text{emp_id, emp_contact} \rightarrow \text{emp_criminal_record}$: This dependency is preserved in the old table

$\text{emp_education} \rightarrow \text{emp_level}$: This dependency is preserved in the new table

$\text{emp_education} \rightarrow \text{emp_department}$: This dependency is preserved in the new table

Thus, both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

$R1 \cap R2 \rightarrow R1 - R2$ or $R1 \cap R2 \rightarrow R2 - R1$ (where R1 is the old table and R2 is the new table)

$R1 \cap R2 = \{ \text{emp_education} \}$

$R1 \cap R2 \rightarrow R1 - R2 = \text{emp_education} \rightarrow \{ \text{emp_id, emp_name, emp_contact, emp_training, emp_criminal_record} \}$

$R1 \cap R2 \rightarrow R2 - R1 = \text{emp_education} \rightarrow \text{emp_department, emp_level}$

Second condition is satisfied and thus this join is said to be lossless with the combination of third and fourth FDs.

13. **sponsor:**

$sponsor_id \rightarrow sponsor_name, sponsor_contact$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: sponsor_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$sponsor_id, sponsor_name, sponsor_contact \rightarrow sponsorship_type$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: sponsor_id, sponsor_name, sponsor_contact) is a super key as sponsor_id, sponsor_name and sponsor_contact can be used to get the other details in other attributes.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

14. **payment_details:**

$payment_ID \rightarrow sponsor_id, payment_Date, payment_type$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: payment_ID) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$payment_ID, sponsor_id \rightarrow amount, payment_Date, payment_type$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: payment_ID, sponsor_id) is a super key as payment_ID and sponsor_id can be used to get the other details in other attributes.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

15. **veterinary:**

$vet_id \rightarrow vet_name, vet_contact, education, work_exp$: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: vet_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

$vet_name, vet_contact \rightarrow work_exp$: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: vet_name, vet_contact) is not a super key and therefore there is a need of a new table. vet_name and vet_contact cannot be used to determine each row in the table.

So, we are splitting this table into 2:

veterinary: (old table)

```

vet_id
vet_name
vet_contact
education
vet_edu (new table)
vet_name
vet_contact
work_exp

```

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created

emp_education \rightarrow emp_department: This dependency is preserved in the new table

vet_id \rightarrow vet_name, vet_contact, education, work_exp: This dependency is preserved in the old table

vet_name, vet_contact \rightarrow work_exp: This dependency is preserved in the new table

Thus, both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

$R1 \cap R2 \rightarrow R1 - R2$ or $R1 \cap R2 \rightarrow R2 - R1$ (where R1 is the old table and R2 is the new table)

$R1 \cap R2 = \{ \text{vet_name}, \text{vet_contact} \}$

$R1 \cap R2 \rightarrow R1 - R2 = \text{vet_name}, \text{vet_contact} \rightarrow \{ \text{vet_id}, \text{education} \}$

$R1 \cap R2 \rightarrow R2 - R1 = \text{vet_name}, \text{vet_contact} \rightarrow \text{work_exp}$

Second condition is satisfied and thus this join is said to be lossless with the combination of third and fourth FDs.

16. animal_transfer_records:

transfer_id \rightarrow transfer_date, animal_id: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: transfer_id) is the primary key which implies it is a super key. As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

transfer_id, animal_id \rightarrow transfer_date, transfer_from, transfer_to: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: transfer_id, animal_id) is a super key as transfer_id, animal_id can be used to get the other details in other attributes. As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

17. employee_payroll:

swipe_id \rightarrow swipe_date, check_in_time, check_out_time, emp_id: This functional dependency satisfies the requirements of BCNF as the determinant (LHS: swipe_id) is the primary key which implies it is a super key.

As we can get the information of all attributes from the attribute mentioned on left side of the relation, we can say that this FD satisfies BCNF.

emp_id -> hourly_pay: This functional dependency does not satisfy the requirements of BCNF as the determinant (LHS: emp_id) is not a super key and therefore there is a need of a new table. emp_id name cannot be used to determine each row in the table as there might be multiple swipes for each employee id.

So, we are splitting this table into 3:

employee_payroll: (old table)

swipe_id

swipe_date

check_in_time

check_out_time

emp_id

employee_pay(new table)

emp_id

hourly_pay

Proof for preservation of Functional Dependencies post-split:

All the functional dependencies either fall into the old or new table created

swipe_id -> swipe_date, check_in_time, check_out_time, emp_id: This dependency is preserved in the old table

emp_id -> hourly_pay: This dependency is preserved in the new table

Thus, both tables are in BCNF

Proof for Lossless Joins:

To ensure that the join is lossless either of the conditions mentioned below should be satisfied:

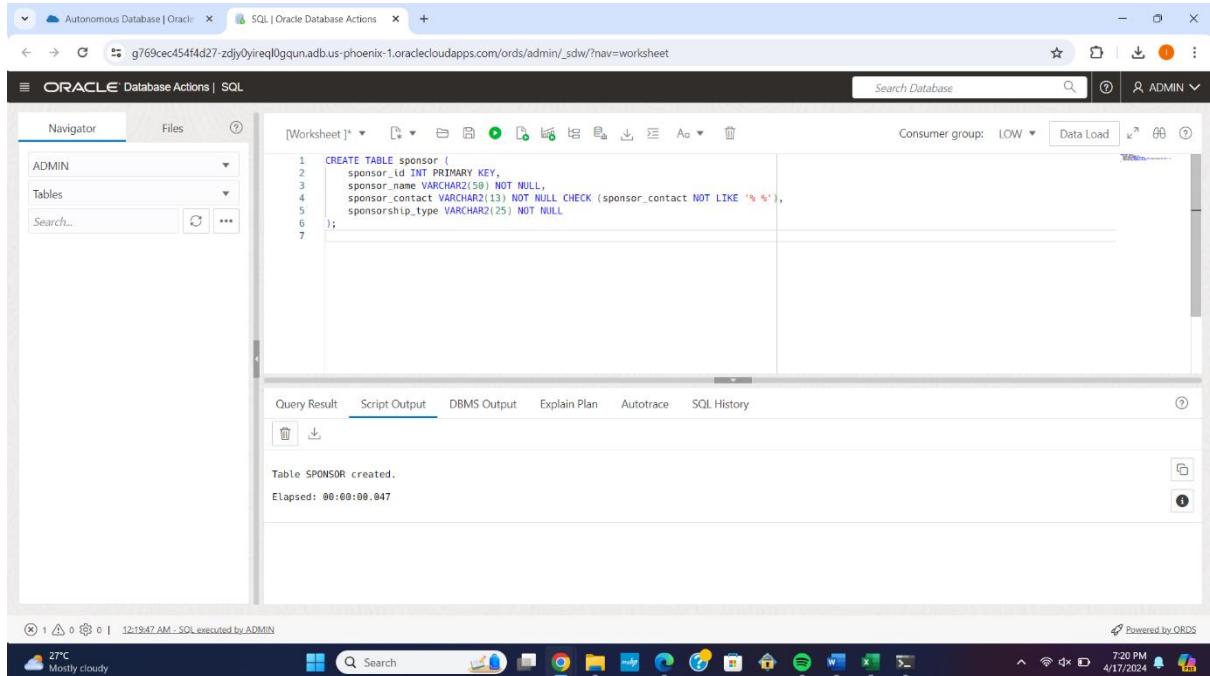
$R1 \cap R2 \rightarrow R1 - R2$ or $R1 \cap R2 \rightarrow R2 - R1$ (where R1 is the old table and R2 is the new table)
 $R1 \cap R2 = \{ \text{emp_id} \}$

$R1 \cap R2 \rightarrow R1 - R2 = \text{emp_id} \rightarrow \{ \text{swipe_id, swipe_date, check_in_time, check_out_time} \}$
 $R1 \cap R2 \rightarrow R2 - R1 = \text{emp_id} \rightarrow \text{hourly_pay}$

Second condition is satisfied and thus this join is said to be lossless with the combination of third and fourth FDs.

Updated Database Screenshots:

Creating a table named sponsor

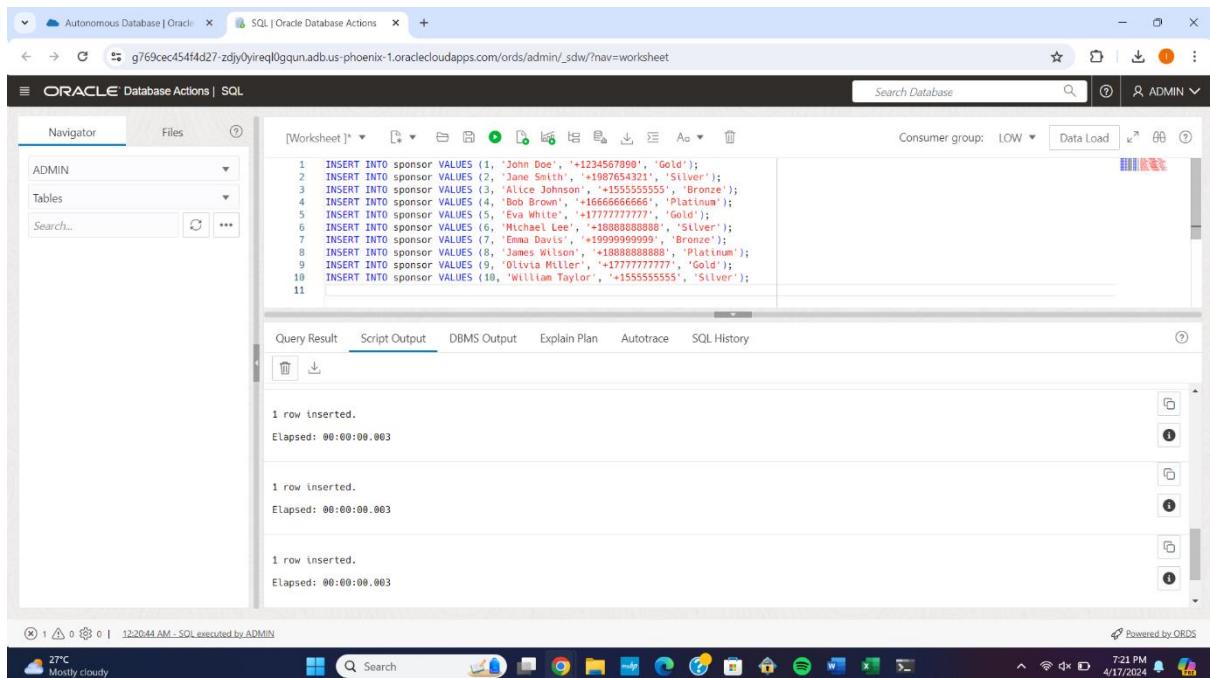


The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Autonomous Database | Oracle' and 'SQL | Oracle Database Actions'. The main workspace is titled '[Worksheet]'. On the left, a 'Navigator' pane shows 'ADMIN' selected under 'Tables'. The central area contains the following SQL code:

```
1 CREATE TABLE sponsor (
2     sponsor_id INT PRIMARY KEY,
3     sponsor_name VARCHAR2(50) NOT NULL,
4     sponsor_contact VARCHAR2(13) NOT NULL CHECK (sponsor_contact NOT LIKE '% %'),
5     sponsorship_type VARCHAR2(25) NOT NULL
6 );
```

Below the code, the 'Query Result' tab is active, displaying the message 'Table SPONSOR created.' and the elapsed time 'Elapsed: 00:00:00.047'. The status bar at the bottom indicates '1 0 0 | 12:19:47 AM - SQL executed by ADMIN'.

Inserting values to sponsor



The screenshot shows the Oracle Database Actions interface. The top navigation bar and workspace are identical to the previous screenshot. The central area contains the following SQL code:

```
1 INSERT INTO sponsor VALUES (1, 'John Doe', '+1234567890', 'Gold');
2 INSERT INTO sponsor VALUES (2, 'Jane Smith', '+1987654321', 'Silver');
3 INSERT INTO sponsor VALUES (3, 'Alice Johnson', '+1555555555', 'Bronze');
4 INSERT INTO sponsor VALUES (4, 'Bob Brown', '+1666666666', 'Platinum');
5 INSERT INTO sponsor VALUES (5, 'Eva White', '+1777777777', 'Gold');
6 INSERT INTO sponsor VALUES (6, 'Michael Lee', '+1888888888', 'Silver');
7 INSERT INTO sponsor VALUES (7, 'Emma Davis', '+1999999999', 'Bronze');
8 INSERT INTO sponsor VALUES (8, 'James Wilson', '+1888888888', 'Platinum');
9 INSERT INTO sponsor VALUES (9, 'Olivia Miller', '+1777777777', 'Gold');
10 INSERT INTO sponsor VALUES (10, 'William Taylor', '+1555555555', 'Silver');
```

The 'Query Result' tab is active, showing three messages: '1 row inserted.', 'Elapsed: 00:00:00.003', and '1 row inserted.', 'Elapsed: 00:00:00.003'. The status bar at the bottom indicates '1 0 0 | 12:20:44 AM - SQL executed by ADMIN'.

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left, the Navigator pane shows 'ADMIN' selected under 'Tables'. The main workspace contains the following SQL code:

```
1 Select * from sponsor;
```

The results pane displays a table with 10 rows of data:

| | SPONSOR_ID | SPONSOR_NAME | SPONSOR_CONTACT | SPONSORSHIP_TYPE |
|----|------------|----------------|-----------------|------------------|
| 1 | 1 | John Doe | +1234567890 | Gold |
| 2 | 2 | Jane Smith | +1987654321 | Silver |
| 3 | 3 | Alice Johnson | +1555555555 | Bronze |
| 4 | 4 | Bob Brown | +1666666666 | Platinum |
| 5 | 5 | Eva White | +1777777777 | Gold |
| 6 | 6 | Michael Lee | +1888888888 | Silver |
| 7 | 7 | Emma Davis | +1999999999 | Bronze |
| 8 | 8 | James Wilson | +1888888888 | Platinum |
| 9 | 9 | Olivia Miller | +1777777777 | Gold |
| 10 | 10 | William Taylor | +1555555555 | Silver |

At the bottom of the results pane, it says 'Execution time: 0.074 seconds'. The status bar at the bottom of the window shows '12:23:24 AM - 10 rows total'.

Creating a table named veterinary

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left, the Navigator pane shows 'ADMIN' selected under 'Tables'. The main workspace contains the following SQL code:

```
1 CREATE TABLE veterinary (
2     vet_id INT PRIMARY KEY,
3     vet_name VARCHAR(59) NOT NULL,
4     vet_contact VARCHAR(13) NOT NULL CHECK (vet_contact NOT LIKE '% %'),
5     education VARCHAR(50),
6     work_exp INT
7 );
8
```

The results pane displays the message 'Table VETERINARY created.' and 'Elapsed: 00:00:00.019'.

At the bottom of the results pane, it says '2024-04-17 12:23:24 AM - SQL executed by ADMIN'. The status bar at the bottom of the window shows '25°C Party cloudy'.

Inserting values to veterinary

```
1  INSERT INTO veterinary VALUES (1, 'Dr. Smith', '+1234567890', 'Doctor of Veterinary Medicine', 10);
2  INSERT INTO veterinary VALUES (2, 'Dr. Johnson', '+1987654321', 'Veterinary Science', 8);
3  INSERT INTO veterinary VALUES (3, 'Dr. Williams', '+1555555555', 'Veterinary Surgery', 12);
4  INSERT INTO veterinary VALUES (4, 'Dr. Brown', '+166666666666', 'Veterinary Pathobiology', 15);
5  INSERT INTO veterinary VALUES (5, 'Dr. Wilson', '+177777777777', 'Veterinary Oncology', 18);
6  INSERT INTO veterinary VALUES (6, 'Dr. Lee', '+188888888888', 'Animal Behavior and Welfare', 7);
7  INSERT INTO veterinary VALUES (7, 'Dr. Martinez', '+199999999999', 'Veterinary Dermatology', 11);
8  INSERT INTO veterinary VALUES (8, 'Dr. White', '+188888888888', 'Veterinary Radiology', 14);
9  INSERT INTO veterinary VALUES (9, 'Dr. Anderson', '+277777777777', 'Small Animal Internal Medicine', 20);
10 INSERT INTO veterinary VALUES (10, 'Dr. Taylor', '+155555555555', 'Equine Medicine', 9);
```

Elapsed: 00:00:00.006
1 row inserted.
Elapsed: 00:00:00.005
1 row inserted.
Elapsed: 00:00:00.005

Creating a table named vet_edu(New)

```
1  CREATE TABLE vet_edu (
2    vet_name VARCHAR2(50) PRIMARY KEY,
3    vet_contact VARCHAR2(13) NOT NULL CHECK (vet_contact NOT LIKE '% %'),
4    work_exp INT
5  );
```

Elapsed: 00:00:00.005
1 row inserted.
Elapsed: 00:00:00.005
Table VET_EDU created.
Elapsed: 00:00:00.019

Inserting values to vet_edu(New)

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

[Worksheet] * + g769cec454fd27-zdjq0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Navigator Files ? [Worksheet] * ▾

ADMIN Tables Search... ...

```

1: INSERT INTO vet_edu VALUES ('Dr. Smith', '+1234567890', 10);
2: INSERT INTO vet_edu VALUES ('Dr. Johnson', '+1987654321', 8);
3: INSERT INTO vet_edu VALUES ('Dr. Williams', '+1555555555', 12);
4: INSERT INTO vet_edu VALUES ('Dr. Brown', '+166666666666', 15);
5: INSERT INTO vet_edu VALUES ('Dr. Wilson', '+177777777777', 18);
6: INSERT INTO vet_edu VALUES ('Dr. Lee', '+188888888888', 7);
7: INSERT INTO vet_edu VALUES ('Dr. Martinez', '+199999999999', 11);
8: INSERT INTO vet_edu VALUES ('Dr. Garcia', '+188888888888', 14);
9: INSERT INTO vet_edu VALUES ('Dr. Anderson', '+177777777777', 20);
10: INSERT INTO vet_edu VALUES ('Dr. Taylor', '+1555555555', 9);
11:

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Elapsed: 00:00:00.005

1 row inserted.

Elapsed: 00:00:00.005

1 row inserted.

Elapsed: 00:00:00.005

Powered by ORDS

25°C Partly cloudy

9:08 PM 4/17/2024

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

[Worksheet] * + g769cec454fd27-zdjq0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Navigator Files ? [Worksheet] * ▾

ADMIN Tables Search... ...

```

1: Select * from vet_edu;
2:

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Download Execution time: 0.016 seconds

| | VET_NAME | VET_CONTACT | WORK_EXP |
|----|--------------|---------------|----------|
| 1 | Dr. Smith | +1234567890 | 10 |
| 2 | Dr. Johnson | +1987654321 | 8 |
| 3 | Dr. Williams | +1555555555 | 12 |
| 4 | Dr. Brown | +166666666666 | 15 |
| 5 | Dr. Wilson | +177777777777 | 18 |
| 6 | Dr. Lee | +188888888888 | 7 |
| 7 | Dr. Martinez | +199999999999 | 11 |
| 8 | Dr. Garcia | +188888888888 | 14 |
| 9 | Dr. Anderson | +177777777777 | 20 |
| 10 | Dr. Taylor | +1555555555 | 9 |

Powered by ORDS

25°C Partly cloudy

9:08 PM 4/17/2024

Creating a table named nutrition

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows 'ADMIN' selected under 'Tables'. The main workspace displays the following SQL code:

```
1 CREATE TABLE nutrition (
2     food_id INT PRIMARY KEY,
3     food_type VARCHAR2(25) NOT NULL,
4     food_quantity INT
5 );
6
```

The bottom pane, 'Query Result', shows the execution results:

```
Elapsed: 00:00:00.003
1 row inserted.
Elapsed: 00:00:00.003
Table NUTRITION created.
Elapsed: 00:00:00.013
```

The status bar at the bottom indicates the command was executed by 'ADMIN' at 12:26:13 AM.

Inserting values to nutrition

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows 'ADMIN' selected under 'Tables'. The main workspace displays the following SQL code:

```
1 INSERT INTO nutrition VALUES (1, 'Dry Food', 100);
2 INSERT INTO nutrition VALUES (2, 'Canned Food', 50);
3 INSERT INTO nutrition VALUES (3, 'Raw Food', 75);
4 INSERT INTO nutrition VALUES (4, 'Wet Food', 80);
5 INSERT INTO nutrition VALUES (5, 'Grain-Free Food', 90);
6 INSERT INTO nutrition VALUES (6, 'Homemade Food', 60);
7 INSERT INTO nutrition VALUES (7, 'Vegetarian Food', 40);
8 INSERT INTO nutrition VALUES (8, 'Organic Food', 70);
9 INSERT INTO nutrition VALUES (9, 'Senior Food', 55);
10 INSERT INTO nutrition VALUES (10, 'Puppy Food', 65);
11
```

The bottom pane, 'Query Result', shows the execution results:

```
Elapsed: 00:00:00.003
1 row inserted.
Elapsed: 00:00:00.003
1 row inserted.
Elapsed: 00:00:00.004
```

The status bar at the bottom indicates the command was executed by 'ADMIN' at 12:26:43 AM.

The screenshot shows the Oracle Database Actions SQL worksheet interface. A query window displays the results of the SQL command: `Select * from nutrition;`. The results are presented in a table with three columns: FOOD_ID, FOOD_TYPE, and FOOD_QUANTITY. The data is as follows:

| FOOD_ID | FOOD_TYPE | FOOD_QUANTITY |
|---------|-----------------|---------------|
| 1 | Dry Food | 100 |
| 2 | Canned Food | 50 |
| 3 | Raw Food | 75 |
| 4 | Wet Food | 80 |
| 5 | Grain-Free Food | 90 |
| 6 | Homemade Food | 60 |
| 7 | Vegetarian Food | 40 |
| 8 | Organic Food | 70 |
| 9 | Senior Food | 55 |
| 10 | Puppy Food | 65 |

The status bar at the bottom indicates "12:34:03 AM - 10 rows total". The system tray shows the date and time as "4/17/2024 7:34 PM".

Creating a table named vaccination

The screenshot shows the Oracle Database Actions SQL worksheet interface. A query window displays the SQL command used to create the `VACCINATION` table:

```
1 CREATE TABLE vaccination (
2     vaccination_id INT PRIMARY KEY,
3     vaccination_name VARCHAR(256) NOT NULL,
4     vaccination_date DATE NOT NULL,
5     vaccination_dosage VARCHAR(25) NOT NULL
6 );
7 
```

The status bar at the bottom indicates "12:34:40 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 7:34 PM".

Inserting values to vaccination

```

1  INSERT INTO vaccination VALUES (1, 'Rabies', TO_DATE('2023-01-15', 'YYYY-MM-DD'), '1st dose');
2  INSERT INTO vaccination VALUES (2, 'Canine Distemper', TO_DATE('2023-02-18', 'YYYY-MM-DD'), 'Booster');
3  INSERT INTO vaccination VALUES (3, 'Feline Calicivirus', TO_DATE('2023-03-05', 'YYYY-MM-DD'), 'Initial');
4  INSERT INTO vaccination VALUES (4, 'Canine Parvovirus', TO_DATE('2023-04-20', 'YYYY-MM-DD'), 'Booster');
5  INSERT INTO vaccination VALUES (5, 'Feline Panleukopenia', TO_DATE('2023-05-12', 'YYYY-MM-DD'), 'Initial');
6  INSERT INTO vaccination VALUES (6, 'Canine Hepatitis', TO_DATE('2023-06-08', 'YYYY-MM-DD'), 'Booster');
7  INSERT INTO vaccination VALUES (7, 'Feline Leukemia', TO_DATE('2023-07-07', 'YYYY-MM-DD'), 'Initial');
8  INSERT INTO vaccination VALUES (8, 'Canine Bordetella', TO_DATE('2023-08-10', 'YYYY-MM-DD'), 'Booster');
9  INSERT INTO vaccination VALUES (9, 'Canine Influenza', TO_DATE('2023-09-03', 'YYYY-MM-DD'), 'Initial');
10 INSERT INTO vaccination VALUES (10, 'Feline Rabies', TO_DATE('2023-10-30', 'YYYY-MM-DD'), 'Booster');

Elapsed: 00:00:00.003
1 row inserted.
Elapsed: 00:00:00.003
1 row inserted.
Elapsed: 00:00:00.003

```

```

1  Select * from vaccination;

```

| VACCINATION_ID | VACCINATION_NAM | VACCINATION_DATE | VACCINATION_DOS |
|----------------|----------------------|-------------------------|-----------------|
| 1 | Rabies | 1/15/2023, 12:00:00 AM | 1st dose |
| 2 | Canine Distemper | 2/10/2023, 12:00:00 AM | Booster |
| 3 | Feline Calicivirus | 3/5/2023, 12:00:00 AM | Initial |
| 4 | Canine Parvovirus | 4/20/2023, 12:00:00 AM | Booster |
| 5 | Feline Panleukopenia | 5/12/2023, 12:00:00 AM | Initial |
| 6 | Canine Hepatitis | 6/8/2023, 12:00:00 AM | Booster |
| 7 | Feline Leukemia | 7/25/2023, 12:00:00 AM | Initial |
| 8 | Canine Bordetella | 8/18/2023, 12:00:00 AM | Booster |
| 9 | Canine Influenza | 9/3/2023, 12:00:00 AM | Initial |
| 10 | Feline Rabies | 10/30/2023, 12:00:00 AM | Booster |

Creating a table named shelter_branch

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the code editor, a CREATE TABLE statement is written:

```
1 CREATE TABLE shelter_branch (
2     shelter_id INT PRIMARY KEY,
3     shelter_name VARCHAR2(50) NOT NULL,
4     shelter_loc VARCHAR2(25) NOT NULL,
5     address VARCHAR2(100) NOT NULL,
6     shelter_contact VARCHAR2(13) NOT NULL CHECK (shelter_contact NOT LIKE '% %')
7 );
8
9
```

The 'Query Result' tab shows the execution results:

- 1 row inserted.
- Elapsed: 00:00:00.003

Below that, another row insertion is shown:

- 1 row inserted.
- Elapsed: 00:00:00.003

Finally, the table creation message is displayed:

- Table SHELTER_BRANCH created.
- Elapsed: 00:00:00.025

The status bar at the bottom indicates the time as 1:31:43 AM - SQL executed by ADMIN.

Inserting values to shelter_branch

The screenshot shows the Oracle Database Actions SQL worksheet interface. A large block of INSERT INTO statements is present in the code editor:

```
1 INSERT INTO shelter_branch VALUES (1, 'Shelter A', 'Location A', 'Address A', '1234567890');
2 INSERT INTO shelter_branch VALUES (2, 'Shelter B', 'Location B', 'Address B', '4321098765');
3 INSERT INTO shelter_branch VALUES (3, 'Shelter C', 'Location C', 'Address C', '3456789012');
4 INSERT INTO shelter_branch VALUES (4, 'Shelter D', 'Location D', 'Address D', '4567899123');
5 INSERT INTO shelter_branch VALUES (5, 'Shelter E', 'Location E', 'Address E', '45678901234');
6 INSERT INTO shelter_branch VALUES (6, 'Shelter F', 'Location F', 'Address F', '6789912345');
7 INSERT INTO shelter_branch VALUES (7, 'Shelter G', 'Location G', 'Address G', '7890123456');
8 INSERT INTO shelter_branch VALUES (8, 'Shelter H', 'Location H', 'Address H', '89901234567');
9 INSERT INTO shelter_branch VALUES (9, 'Shelter I', 'Location I', 'Address I', '9012345679');
10 INSERT INTO shelter_branch VALUES (10, 'Shelter J', 'Location J', 'Address J', '1234567890');
```

The 'Query Result' tab shows the execution results for each insert statement:

- 1 row inserted.
- Elapsed: 00:00:00.006

- 1 row inserted.
- Elapsed: 00:00:00.005

- 1 row inserted.
- Elapsed: 00:00:00.007

The status bar at the bottom indicates the time as 1:32:58 AM - SQL executed by ADMIN.

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left, the Navigator pane shows 'ADMIN' selected under 'Tables'. The main area displays a query result for the 'shelter_branch' table:

```
1 Select * from shelter_branch;
```

| SHELTER_ID | SHELTER_NAME | SHELTER_LOC | ADDRESS | SHELTER_CONTACT |
|------------|--------------|-------------|-----------|-----------------|
| 1 | Shelter A | Location A | Address A | +1234567890 |
| 2 | Shelter B | Location B | Address B | +2345678901 |
| 3 | Shelter C | Location C | Address C | +3456789012 |
| 4 | Shelter D | Location D | Address D | +4567890123 |
| 5 | Shelter E | Location E | Address E | +5678901234 |
| 6 | Shelter F | Location F | Address F | +6789012345 |
| 7 | Shelter G | Location G | Address G | +7890123456 |
| 8 | Shelter H | Location H | Address H | +8901234567 |
| 9 | Shelter I | Location I | Address I | +9012345678 |
| 10 | Shelter J | Location J | Address J | +0123456789 |

At the bottom, it says '1:34:25 AM - 10 rows total'. The system tray shows the date as 4/17/2024.

Creating a table named shelter_loc_details(New)

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left, the Navigator pane shows 'ADMIN' selected under 'Tables'. The main area displays the creation of a new table:

```
1 CREATE TABLE shelter_loc_details (
2     shelter_loc VARCHAR2(25) PRIMARY KEY,
3     pincode CHAR(6) NOT NULL CHECK (LENGTH(pincode) = 6)
4 );
5
6
7
```

The 'Query Result' tab shows three rows inserted:

- 1 row inserted.
Elapsed: 00:00:00.006
- 1 row inserted.
Elapsed: 00:00:00.005
- 1 row inserted.
Elapsed: 00:00:00.007

At the bottom, it says '1:35:18 AM - SQL executed by ADMIN'. The system tray shows the date as 4/17/2024.

Inserting values to shelter_loc_details(New)

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```

1  INSERT INTO shelter_loc_details VALUES ('Location A', '123456');
2  INSERT INTO shelter_loc_details VALUES ('Location B', '234567');
3  INSERT INTO shelter_loc_details VALUES ('Location C', '345678');
4  INSERT INTO shelter_loc_details VALUES ('Location D', '456789');
5  INSERT INTO shelter_loc_details VALUES ('Location E', '567890');
6  INSERT INTO shelter_loc_details VALUES ('Location F', '678991');
7  INSERT INTO shelter_loc_details VALUES ('Location G', '7890123');
8  INSERT INTO shelter_loc_details VALUES ('Location H', '8901234');
9  INSERT INTO shelter_loc_details VALUES ('Location I', '901234');
10 INSERT INTO shelter_loc_details VALUES ('Location J', '012345');
11

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

```

1 row inserted.
Elapsed: 00:00:00.004

1 row inserted.
Elapsed: 00:00:00.005

1 row inserted.
Elapsed: 00:00:00.005

```

Powered by ORDS

Upcoming Earnings

1:35:55 AM - SQL executed by ADMIN

Powered by ORDS

Upcoming Earnings

1:36:27 AM - 10 rows total

Powered by ORDS

Creating a table named animal_features

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows the ADMIN schema with Tables selected. The main area contains the following SQL code:

```
1 CREATE TABLE animal_features (
2     animal_features_id INT PRIMARY KEY,
3     a_color VARCHAR2(50) NOT NULL,
4     a_weight DECIMAL NOT NULL,
5     a_height DECIMAL NOT NULL,
6     a_eye_color VARCHAR2(50) NOT NULL,
7     a_condition VARCHAR2(25) NOT NULL
8 );
9
```

Below the code, the Query Result tab shows the output:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

Table ANIMAL_FEATURES created.
Elapsed: 00:00:00.017

The status bar at the bottom indicates "12:36:19 AM - SQL executed by ADMIN".

Inserting values to animal_features

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows the ADMIN schema with Tables selected. The main area contains the following SQL code:

```
1 INSERT INTO animal_features VALUES (1, 'Brown', 12.5, 30.5, 'Blue', 'Healthy');
2 INSERT INTO animal_features VALUES (2, 'Black', 20.2, 35.8, 'Brown', 'Underweight');
3 INSERT INTO animal_features VALUES (3, 'White', 8.8, 25.8, 'Green', 'Healthy');
4 INSERT INTO animal_features VALUES (4, 'Gray', 15.9, 28.0, 'Amer', 'Overweight');
5 INSERT INTO animal_features VALUES (5, 'Golden', 18.9, 32.2, 'Hazel', 'Healthy');
6 INSERT INTO animal_features VALUES (6, 'Spotted', 10.7, 26.8, 'Gray', 'Underweight');
7 INSERT INTO animal_features VALUES (7, 'Tabby', 14.3, 29.5, 'Yellow', 'Healthy');
8 INSERT INTO animal_features VALUES (8, 'Orange', 16.8, 31.6, 'Blue', 'Overweight');
9 INSERT INTO animal_features VALUES (9, 'Tan', 11.2, 27.8, 'Green', 'Healthy');
10 INSERT INTO animal_features VALUES (10, 'Calico', 9.5, 24.8, 'Brown', 'Underweight');
```

Below the code, the Query Result tab shows the output:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

```
1 row inserted.  
Elapsed: 00:00:00.003
```

```
1 row inserted.  
Elapsed: 00:00:00.003
```

The status bar at the bottom indicates "12:36:52 AM - SQL executed by ADMIN".

The screenshot shows the Oracle Database Actions interface with a SQL worksheet titled '[Worksheet]'. The query executed is:

```
1 Select * from animal_features;
```

The results are displayed in a table:

| | ANIMAL_FEATURES_ID | A_COLOR | A_WEIGHT | A_HEIGHT | A_EYE_COLOR | A_CONDITION |
|----|--------------------|---------|----------|----------|-------------|-------------|
| 1 | | Brown | 13 | 31 | Blue | Healthy |
| 2 | | Black | 20 | 35 | Brown | Underweight |
| 3 | | White | 8 | 25 | Green | Healthy |
| 4 | | Gray | 15 | 28 | Amber | Overweight |
| 5 | | Golden | 19 | 32 | Hazel | Healthy |
| 6 | | Spotted | 11 | 27 | Gray | Underweight |
| 7 | | Tabby | 14 | 30 | Yellow | Healthy |
| 8 | | Orange | 17 | 31 | Blue | Overweight |
| 9 | | Tan | 11 | 27 | Green | Healthy |
| 10 | | Calico | 10 | 25 | Brown | Underweight |

Execution time: 0.012 seconds

Creating a table named facilities

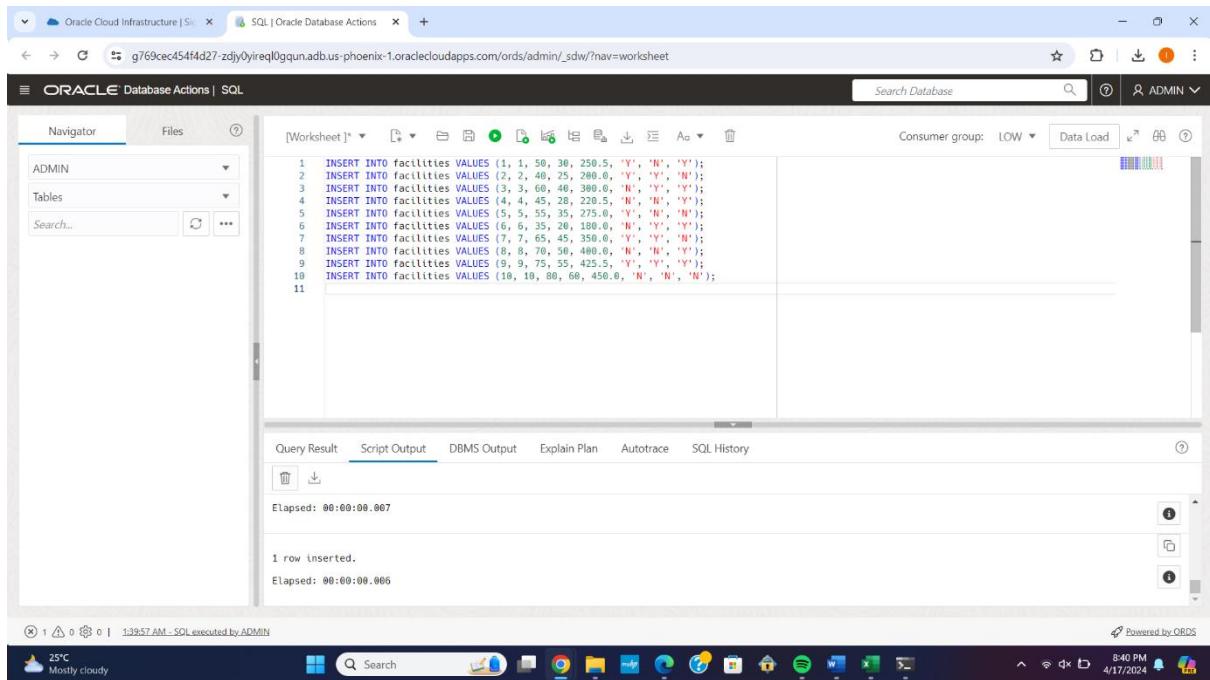
The screenshot shows the Oracle Database Actions interface with a SQL worksheet titled '[Worksheet]'. The query executed is:

```
1 CREATE TABLE facilities (
2     facilities_id INT PRIMARY KEY,
3     shelter_id INT REFERENCES shelter_branch(shelter_id),
4     f_capacity INT NOT NULL,
5     f_anl_count INT NOT NULL,
6     f_area DECIMAL NOT NULL,
7     play_areas CHAR(1) NOT NULL,
8     temp_control CHAR(1) NOT NULL,
9     outdoor_space CHAR(1) NOT NULL
10 );
11
12
```

The results show the table was created successfully:

```
Elapsed: 00:00:00.005
Table FACILITIES created.
Elapsed: 00:00:00.025
```

Inserting values to facilities



The screenshot shows the Oracle Database Actions SQL worksheet interface. The code entered is:

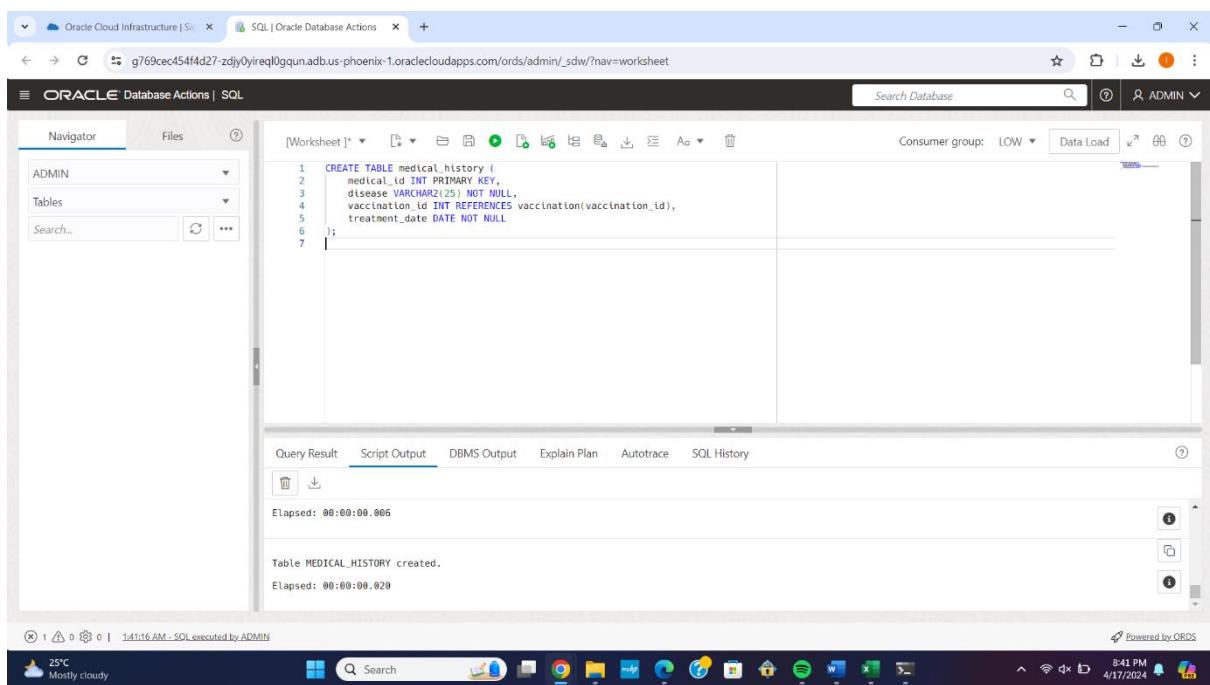
```
1 INSERT INTO facilities VALUES (1, 1, 50, 30, 250.5, 'Y', 'N', 'Y');
2 INSERT INTO facilities VALUES (2, 2, 40, 25, 280.0, 'Y', 'Y', 'N');
3 INSERT INTO facilities VALUES (3, 3, 60, 40, 380.0, 'N', 'Y', 'Y');
4 INSERT INTO facilities VALUES (4, 4, 45, 20, 220.5, 'N', 'N', 'Y');
5 INSERT INTO facilities VALUES (5, 5, 55, 35, 275.0, 'Y', 'N', 'N');
6 INSERT INTO facilities VALUES (6, 6, 35, 20, 180.0, 'N', 'Y', 'Y');
7 INSERT INTO facilities VALUES (7, 7, 65, 45, 380.0, 'N', 'N', 'N');
8 INSERT INTO facilities VALUES (8, 8, 70, 50, 400.0, 'N', 'N', 'N');
9 INSERT INTO facilities VALUES (9, 9, 75, 55, 425.0, 'Y', 'Y', 'Y');
10 INSERT INTO facilities VALUES (10, 10, 80, 60, 450.0, 'N', 'N', 'N');
```

The results section shows:

- Elapsed: 00:00:00.007
- 1 row inserted.
- Elapsed: 00:00:00.006

At the bottom, it says "Powered by ORDS".

Creating a table named medical_history



The screenshot shows the Oracle Database Actions SQL worksheet interface. The code entered is:

```
1 CREATE TABLE medical_history (
2     medical_id INT PRIMARY KEY,
3     disease VARCHAR(25) NOT NULL,
4     vaccination_id INT REFERENCES vaccination(vaccination_id),
5     treatment_date DATE NOT NULL
6 );
```

The results section shows:

- Elapsed: 00:00:00.006
- Table MEDICAL_HISTORY created.
- Elapsed: 00:00:00.020

At the bottom, it says "Powered by ORDS".

Inserting values to medical_history

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```

1  INSERT INTO medical_history VALUES (1, 'Fever', 1, DATE '2023-05-10');
2  INSERT INTO medical_history VALUES (2, 'Cough', 2, DATE '2023-07-15');
3  INSERT INTO medical_history VALUES (3, 'Infection', 3, DATE '2023-08-20');
4  INSERT INTO medical_history VALUES (4, 'Broken Leg', NULL, DATE '2023-09-25');
5  INSERT INTO medical_history VALUES (5, 'Allergy', 4, DATE '2023-10-05');
6  INSERT INTO medical_history VALUES (6, 'Flu', 5, DATE '2023-11-05');
7  INSERT INTO medical_history VALUES (7, 'Ear Infection', 6, DATE '2023-12-10');
8  INSERT INTO medical_history VALUES (8, 'Skin Rash', NULL, DATE '2024-01-15');
9  INSERT INTO medical_history VALUES (9, 'Arthritis', 7, DATE '2024-02-20');
10 INSERT INTO medical_history VALUES (10, 'Diabetes', 8, DATE '2024-03-25');
11

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Elapsed: 00:00:00.005

1 row inserted.

Elapsed: 00:00:00.005

Powered by ORDS

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```

1  Select * from medical_history;

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

| MEDICAL_ID | DISEASE | VACCINATION_ID | TREATMENT_DATE |
|------------|---------------|----------------|-------------------------|
| 1 | Fever | 1 | 5/10/2023, 12:00:00 AM |
| 2 | Cough | 2 | 7/15/2023, 12:00:00 AM |
| 3 | Infection | 3 | 8/20/2023, 12:00:00 AM |
| 4 | Broken Leg | (null) | 9/25/2023, 12:00:00 AM |
| 5 | Allergy | 4 | 10/30/2023, 12:00:00 AM |
| 6 | Flu | 5 | 11/5/2023, 12:00:00 AM |
| 7 | Ear Infection | 6 | 12/10/2023, 12:00:00 AM |
| 8 | Skin Rash | (null) | 1/15/2024, 12:00:00 AM |
| 9 | Arthritis | 7 | 2/20/2024, 12:00:00 AM |
| 10 | Diabetes | 8 | 3/25/2024, 12:00:00 AM |

10 rows total

Powered by ORDS

Creating a table named disease_mitigation (New)

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left corner, there's a browser tab for 'Oracle Cloud Infrastructure | SQL' and another for 'SQL | Oracle Database Actions'. The main window title is 'ORACLE Database Actions | SQL'. The left sidebar has a 'Navigator' section with 'ADMIN' selected, showing 'Tables' and a search bar. The central workspace contains the following SQL code:

```
1 CREATE TABLE disease_mitigation (
2     disease VARCHAR2(25) PRIMARY KEY,
3     vaccination_id INT REFERENCES vaccination(vaccination_id)
4 );
5
```

Below the code, the 'Script Output' tab is active, displaying the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.005
```

Further down, it shows the creation of the table:

```
Table DISEASE_MITIGATION created.  
Elapsed: 00:00:00.020
```

The bottom status bar indicates '1:44:15 AM - SQL executed by ADMIN'.

Inserting values to disease_mitigation (New)

This screenshot shows the same Oracle Database Actions SQL worksheet interface. The central workspace contains the following SQL code for inserting multiple rows into the 'disease_mitigation' table:

```
1 INSERT INTO disease_mitigation VALUES ('Fever', 1);
2 INSERT INTO disease_mitigation VALUES ('Cough', 2);
3 INSERT INTO disease_mitigation VALUES ('Infection', 3);
4 INSERT INTO disease_mitigation VALUES ('Broken Leg', NULL);
5 INSERT INTO disease_mitigation VALUES ('Allergy', 4);
6 INSERT INTO disease_mitigation VALUES ('Flu', 5);
7 INSERT INTO disease_mitigation VALUES ('Ear Infection', 6);
8 INSERT INTO disease_mitigation VALUES ('Skin Rash', NULL);
9 INSERT INTO disease_mitigation VALUES ('Arthritis', 7);
10 INSERT INTO disease_mitigation VALUES ('Diabetes', 8);
11
```

Below the code, the 'Script Output' tab is active, showing the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.007
```

Further down, it shows another row being inserted:

```
1 row inserted.  
Elapsed: 00:00:00.008
```

The bottom status bar indicates '1:47:43 AM - SQL executed by ADMIN'.

```

1  Select * from disease_mitigation;

```

| | DISEASE | VACCINATION_ID |
|----|---------------|----------------|
| 1 | Fever | 1 |
| 2 | Cough | 2 |
| 3 | Infection | 3 |
| 4 | Broken Leg | (null) |
| 5 | Allergy | 4 |
| 6 | Flu | 5 |
| 7 | Ear Infection | 6 |
| 8 | Skin Rash | (null) |
| 9 | Arthritis | 7 |
| 10 | Diabetes | 8 |

1 2 3 4 5 6 7 8 9 10 | 1:45:08 AM - 10 rows total

Creating a table named adoption_details

```

1 CREATE TABLE adoption_details (
2     ad_id INT PRIMARY KEY,
3     payment_id INT REFERENCES payment_details(payment_id),
4     ad_name VARCHAR2(50) NOT NULL,
5     ad_contact VARCHAR2(13) NOT NULL CHECK (ad_contact NOT LIKE '%-%'),
6     ad_date DATE NOT NULL,
7     ad_fee NUMERIC NOT NULL
8 );
9
10
11

```

1 row inserted.
Elapsed: 00:00:00.003

Table ADOPTION_DETAILS created.
Elapsed: 00:00:00.025

1 2 3 4 5 6 7 8 9 10 11 | 12:44:21 AM - SQL executed by ADMIN

Inserting values to adoption_details

Autonomous Database | Oracle | SQL | Oracle Database Actions

```
[Worksheet]*
1 INSERT INTO adoption_details VALUES (1, 1, 'John Doe', '+1234567890', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 500.00);
2 INSERT INTO adoption_details VALUES (2, 2, 'Jane Smith', '+1987654321', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 750.00);
3 INSERT INTO adoption_details VALUES (3, 3, 'Michael Johnson', '+1765432890', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 1000.00);
4 INSERT INTO adoption_details VALUES (4, 4, 'Emily Davis', '+1456789234', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 600.00);
5 INSERT INTO adoption_details VALUES (5, 5, 'William Brown', '+1980765432', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 900.00);
6 INSERT INTO adoption_details VALUES (6, 6, 'Olivia Wilson', '+1876543210', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 1200.00);
7 INSERT INTO adoption_details VALUES (7, 7, 'Ethan Martinez', '+1234987654', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 800.00);
8 INSERT INTO adoption_details VALUES (8, 8, 'Ava Taylor', '+1654321987', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 950.00);
9 INSERT INTO adoption_details VALUES (9, 9, 'Noah Anderson', '+1987654321', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 1100.00);
10 INSERT INTO adoption_details VALUES (10, 10, 'Emma Garcia', '+1765432890', TO_DATE('2024-02-22', 'YYYY-MM-DD'), 850.00);
```

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.004

1 0 0 0 | 12:45:51 AM - SQL executed by ADMIN

Powered by ORDS

27°C
Mostly cloudy

7:44 PM
4/17/2024

Autonomous Database | Oracle | SQL | Oracle Database Actions

```
[Worksheet]*
1 Select * from adoption_details;
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

| AD_ID | PAYMENT_ID | AD_NAME | AD_CONTACT | AD_DATE | AD_FEE |
|-------|------------|-----------------|-------------|-----------------------|--------|
| 1 | 1 | John Doe | +1234567890 | 2/22/2024, 12:00:00 A | 500 |
| 2 | 2 | Jane Smith | +1987654321 | 2/22/2024, 12:00:00 A | 750 |
| 3 | 3 | Michael Johnson | +1765432890 | 2/22/2024, 12:00:00 A | 1000 |
| 4 | 4 | Emily Davis | +1456789234 | 2/22/2024, 12:00:00 A | 600 |
| 5 | 5 | William Brown | +1980765432 | 2/22/2024, 12:00:00 A | 900 |
| 6 | 6 | Olivia Wilson | +1876543210 | 2/22/2024, 12:00:00 A | 1200 |
| 7 | 7 | Ethan Martinez | +1234987654 | 2/22/2024, 12:00:00 A | 800 |
| 8 | 8 | Ava Taylor | +1654321987 | 2/22/2024, 12:00:00 A | 950 |
| 9 | 9 | Noah Anderson | +1987654321 | 2/22/2024, 12:00:00 A | 1100 |
| 10 | 10 | Emma Garcia | +1765432890 | 2/22/2024, 12:00:00 A | 850 |

1 0 0 0 | 12:45:55 AM - 10 rows total

27°C
Mostly cloudy

7:45 PM
4/17/2024

Creating a table named surrender_details

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left corner, there are tabs for 'Autonomous Database | Oracle' and 'SQL | Oracle Database Actions'. The main area displays the following SQL code:

```
1 CREATE TABLE surrender_details (
2     s_id INT PRIMARY KEY,
3     s_date DATE NOT NULL,
4     s_name VARCHAR2(50) NOT NULL,
5     s_reason VARCHAR2(200) NOT NULL
6 );
7
```

Below the code, the message 'Table SURRENDER_DETAILS created.' is displayed. The status bar at the bottom left shows '12:46:10 AM - SQL executed by ADMIN'. The system tray at the bottom right indicates it's 7:46 PM on 4/17/2024, with a weather icon showing 27°C and mostly cloudy.

Inserting values to surrender_details

The screenshot shows the Oracle Database Actions SQL worksheet interface. The top-left tab is 'SQL | Oracle Database Actions'. The main area displays the following SQL code for inserting 18 rows into the SURRENDER_DETAILS table:

```
1 INSERT INTO surrender_details VALUES (1, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'John Doe', 'Moving to another country');
2 INSERT INTO surrender_details VALUES (2, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Jane Smith', 'Allergic reaction to pet');
3 INSERT INTO surrender_details VALUES (3, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Michael Johnson', 'Financial difficulties');
4 INSERT INTO surrender_details VALUES (4, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Emily Davis', 'Change in living situation');
5 INSERT INTO surrender_details VALUES (5, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'William Brown', 'Owner passed away');
6 INSERT INTO surrender_details VALUES (6, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Olivia Wilson', 'Moving to a place where pets are not allowed');
7 INSERT INTO surrender_details VALUES (7, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Ethan Martinez', 'Health issues of owner');
8 INSERT INTO surrender_details VALUES (8, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Ava Taylor', 'Unable to afford pet care');
9 INSERT INTO surrender_details VALUES (9, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Noah Anderson', 'Owner no longer interested in pet');
10 INSERT INTO surrender_details VALUES (10, TO_DATE('2024-02-22', 'YYYY-MM-DD'), 'Emma Garcia', 'Behavioral problems with pet');
```

Below the code, the message '1 row inserted.' is displayed twice, indicating the successful insertion of 18 rows. The status bar at the bottom left shows '12:46:50 AM - SQL executed by ADMIN'. The system tray at the bottom right indicates it's 7:46 PM on 4/17/2024, with a weather icon showing 27°C and mostly cloudy.

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Autonomous Database | Oracle' and 'SQL | Oracle Database Actions'. The URL in the address bar is g769cec454fd27-zdjj0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet. The main area displays a SQL query: `1 Select * from surrender_details;`. Below the query is a table with 10 rows of data:

| S_ID | S_DATE | S_NAME | S_REASON |
|------|---------------------|-------------------|-------------------------|
| 1 | 2/22/2024, 12:00:00 | A John Doe | Moving to another cc |
| 2 | 2/22/2024, 12:00:00 | A Jane Smith | Allergic reaction to pi |
| 3 | 2/22/2024, 12:00:00 | A Michael Johnson | Financial difficulties |
| 4 | 2/22/2024, 12:00:00 | A Emily Davis | Change in living situa |
| 5 | 2/22/2024, 12:00:00 | A William Brown | Owner passed away |
| 6 | 2/22/2024, 12:00:00 | A Olivia Wilson | Moving to a place wh |
| 7 | 2/22/2024, 12:00:00 | A Ethan Martinez | Health issues of owne |
| 8 | 2/22/2024, 12:00:00 | A Ava Taylor | Unable to afford pet i |
| 9 | 2/22/2024, 12:00:00 | A Noah Anderson | Owner no longer int |
| 10 | 2/22/2024, 12:00:00 | A Emma Garcia | Behavioral problems i |

At the bottom of the interface, there is a status bar showing '12:47:17 AM - 10 rows total' and a weather widget indicating '27°C Mostly cloudy'. The system tray shows the date and time as '7:47 PM 4/17/2024'.

Creating a table named payment_details

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Autonomous Database | Oracle' and 'SQL | Oracle Database Actions'. The URL in the address bar is g769cec454fd27-zdjj0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet. The main area displays a SQL script to create a table:

```
1 CREATE TABLE payment_details (
2     payment_id INT PRIMARY KEY,
3     sponsor_id INT REFERENCES sponsor(sponsor_id),
4     amount NUMERIC NOT NULL,
5     payment_date DATE NOT NULL,
6     payment_type VARCHAR2(25) NOT NULL
7 );
8 
```

Below the script, the message 'Table PAYMENT_DETAILS created.' is displayed. The status bar at the bottom shows 'Elapsed: 00:00:00.034' and the date and time as '12:41:55 AM - SQL executed by ADMIN'.

Inserting values to payment_details

Autonomous Database | Oracle | SQL | Oracle Database Actions

[Worksheet] * +

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Administrator

Tables

Search...

```

1 INSERT INTO payment_details VALUES (1, 1, 500.00, DATE '2023-01-10', 'Online');
2 INSERT INTO payment_details VALUES (2, 2, 750.00, DATE '2023-02-15', 'Cash');
3 INSERT INTO payment_details VALUES (3, 3, 600.00, DATE '2023-03-20', 'Credit Card');
4 INSERT INTO payment_details VALUES (4, 4, 450.00, DATE '2023-04-25', 'Online');
5 INSERT INTO payment_details VALUES (5, 5, 700.00, DATE '2023-05-30', 'Cash');
6 INSERT INTO payment_details VALUES (6, 6, 800.00, DATE '2023-06-05', 'Credit Card');
7 INSERT INTO payment_details VALUES (7, 7, 550.00, DATE '2023-07-10', 'Online');
8 INSERT INTO payment_details VALUES (8, 8, 650.00, DATE '2023-08-15', 'Cash');
9 INSERT INTO payment_details VALUES (9, 9, 650.00, DATE '2023-09-20', 'Credit Card');
10 INSERT INTO payment_details VALUES (10, 10, 750.00, DATE '2023-10-25', 'Online');
11

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

Powered by ORDS

12:42:35 AM - SQL executed by ADMIN

27°C Mostly cloudy

7:42 PM 4/17/2024

Autonomous Database | Oracle | SQL | Oracle Database Actions

[Worksheet] * +

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Administrator

Tables

Search...

```

1 Select * from payment_details;
2

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Download Execution time: 0.02 seconds

| PAYMENT_ID | SPONSOR_ID | AMOUNT | PAYMENT_DATE | PAYMENT_TYPE |
|------------|------------|--------|--------------|-----------------------------------|
| 1 | 1 | 1 | 500 | 1/10/2023, 12:00:00 A Online |
| 2 | 2 | 2 | 750 | 2/15/2023, 12:00:00 A Cash |
| 3 | 3 | 3 | 600 | 3/20/2023, 12:00:00 A Credit Card |
| 4 | 4 | 4 | 450 | 4/25/2023, 12:00:00 A Online |
| 5 | 5 | 5 | 700 | 5/30/2023, 12:00:00 A Cash |
| 6 | 6 | 6 | 800 | 6/5/2023, 12:00:00 A Credit Card |
| 7 | 7 | 7 | 550 | 7/10/2023, 12:00:00 A Online |
| 8 | 8 | 8 | 900 | 8/15/2023, 12:00:00 A Cash |
| 9 | 9 | 9 | 650 | 9/20/2023, 12:00:00 A Credit Card |
| 10 | 10 | 10 | 750 | 10/25/2023, 12:00:00 Online |

12:43:04 AM - 10 rows total

27°C Mostly cloudy

7:43 PM 4/17/2024

Creating a table named training_history

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows 'ADMIN' selected under 'Tables'. The main workspace contains the following SQL code:

```
1 CREATE TABLE training_history (
2     training_id INT PRIMARY KEY,
3     trainer_name VARCHAR2(50) NOT NULL,
4     training_date DATE NOT NULL,
5     training_type VARCHAR2(25) NOT NULL,
6     training_stage VARCHAR2(25) NOT NULL
7 );
```

Below the code, the 'Query Result' tab displays the output:

```
1 row inserted.  
Elapsed: 00:00:00.005
```

Table TRAINING_HISTORY created.
Elapsed: 00:00:00.019

The status bar at the bottom indicates the command was executed at 12:47:54 AM by ADMIN.

Inserting values to training_history

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows 'ADMIN' selected under 'Tables'. The main workspace contains the following SQL code:

```
1 INSERT INTO training_history VALUES (1, 'John Smith', TO_DATE('2023-05-10', 'YYYY-MM-DD'), 'Basic Obedience', 'Beginner');
2 INSERT INTO training_history VALUES (2, 'Emily Johnson', TO_DATE('2023-06-15', 'YYYY-MM-DD'), 'Agility Training', 'Intermediate');
3 INSERT INTO training_history VALUES (3, 'Michael Brown', TO_DATE('2023-07-20', 'YYYY-MM-DD'), 'Behavior Modification', 'Advanced');
4 INSERT INTO training_history VALUES (4, 'Sophia Davis', TO_DATE('2023-08-25', 'YYYY-MM-DD'), 'Service Dog Training', 'Beginner');
5 INSERT INTO training_history VALUES (5, 'Matthew Wilson', TO_DATE('2023-09-30', 'YYYY-MM-DD'), 'Puppy Socialization', 'Intermediate');
6 INSERT INTO training_history VALUES (6, 'Olivia Garcia', TO_DATE('2023-10-05', 'YYYY-MM-DD'), 'Aggression Management', 'Advanced');
7 INSERT INTO training_history VALUES (7, 'Daniel Taylor', TO_DATE('2023-11-10', 'YYYY-MM-DD'), 'Therapy Dog Training', 'Beginner');
8 INSERT INTO training_history VALUES (8, 'Ava Martinez', TO_DATE('2023-12-15', 'YYYY-MM-DD'), 'Trick Training', 'Intermediate');
9 INSERT INTO training_history VALUES (9, 'Noah Anderson', TO_DATE('2024-01-20', 'YYYY-MM-DD'), 'Canine Good Citizen', 'Advanced');
10 INSERT INTO training_history VALUES (10, 'Emma Hernandez', TO_DATE('2024-02-25', 'YYYY-MM-DD'), 'Search and Rescue', 'Beginner');
```

Below the code, the 'Query Result' tab displays the output:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

1 row inserted.
Elapsed: 00:00:00.003

The status bar at the bottom indicates the command was executed at 12:48:33 AM by ADMIN.

The screenshot shows the Oracle Database Actions interface with a SQL worksheet open. The query executed is:

```
1  Select * from training_history;
```

The resulting table contains 10 rows of training history data:

| | TRAINING_ID | TRAINER_NAME | TRAINING_DATE | TRAINING_TYPE | TRAINING_STAGE |
|----|-------------|----------------|------------------------|-----------------------|----------------|
| 1 | 1 | John Smith | 5/10/2023, 12:00:00 A | Basic Obedience | Beginner |
| 2 | 2 | Emily Johnson | 6/15/2023, 12:00:00 A | Agility Training | Intermediate |
| 3 | 3 | Michael Brown | 7/20/2023, 12:00:00 A | Behavior Modification | Advanced |
| 4 | 4 | Sophia Davis | 8/25/2023, 12:00:00 A | Service Dog Training | Beginner |
| 5 | 5 | Matthew Wilson | 9/30/2023, 12:00:00 A | Puppy Socialization | Intermediate |
| 6 | 6 | Olivia Garcia | 10/5/2023, 12:00:00 A | Aggression Management | Advanced |
| 7 | 7 | Daniel Taylor | 11/10/2023, 12:00:00 A | Therapy Dog Training | Beginner |
| 8 | 8 | Ava Martinez | 12/15/2023, 12:00:00 A | Trick Training | Intermediate |
| 9 | 9 | Noah Anderson | 1/20/2024, 12:00:00 A | Canine Good Citizen | Advanced |
| 10 | 10 | Emma Hernandez | 2/25/2024, 12:00:00 A | Search and Rescue | Beginner |

Creating a table named grooming_history

The screenshot shows the Oracle Database Actions interface with a SQL worksheet open. The query executed is:

```
1 CREATE TABLE grooming_history (
2   grooming_id INT PRIMARY KEY,
3   groomer_name VARCHAR2(50) NOT NULL,
4   grooming_date DATE NOT NULL,
5   grooming_type VARCHAR2(25) NOT NULL
6 );
7 ;
```

The resulting output indicates the table was created successfully:

```
1 row inserted.  
Elapsed: 00:00:00.003  
  
Table GROOMING_HISTORY created.  
Elapsed: 00:00:00.015
```

Inserting values to grooming_history

Autonomous Database | Oracle | SQL | Oracle Database Actions

[Worksheet] * +

g769cec454fd27-zdjy0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Navigator Files ?

ADMIN Tables Search...

```

1 INSERT INTO grooming_history VALUES (1, 'Mary Johnson', TO_DATE('2023-05-10', 'YYYY-MM-DD'), 'Bath and Brush');
2 INSERT INTO grooming_history VALUES (2, 'David Wilson', TO_DATE('2023-06-15', 'YYYY-MM-DD'), 'Haircut');
3 INSERT INTO grooming_history VALUES (3, 'Jessica Thompson', TO_DATE('2023-07-20', 'YYYY-MM-DD'), 'Nail Trim');
4 INSERT INTO grooming_history VALUES (4, 'Christopher Martinez', TO_DATE('2023-08-25', 'YYYY-MM-DD'), 'Ear Cleaning');
5 INSERT INTO grooming_history VALUES (5, 'Jennifer Davis', TO_DATE('2023-09-30', 'YYYY-MM-DD'), 'Full Groom');
6 INSERT INTO grooming_history VALUES (6, 'Ryan Garcia', TO_DATE('2023-10-05', 'YYYY-MM-DD'), 'De-shedding Treatment');
7 INSERT INTO grooming_history VALUES (7, 'Emma Hernandez', TO_DATE('2023-10-10', 'YYYY-MM-DD'), 'Teeth Cleaning');
8 INSERT INTO grooming_history VALUES (8, 'Matthew Taylor', TO_DATE('2023-12-15', 'YYYY-MM-DD'), 'Flea and Tick Treatment');
9 INSERT INTO grooming_history VALUES (9, 'Olivia Anderson', TO_DATE('2024-01-29', 'YYYY-MM-DD'), 'Paw Trim');
10 INSERT INTO grooming_history VALUES (10, 'Sophia Brown', TO_DATE('2024-02-25', 'YYYY-MM-DD'), 'Sanitary Trim');

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

Powered by ORDS

1 0 0 0 | 12:50:21 AM - SQL executed by ADMIN

27°C Mostly cloudy

Search

7:50 PM 4/17/2024

Autonomous Database | Oracle | SQL | Oracle Database Actions

[Worksheet] * +

g769cec454fd27-zdjy0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet

ORACLE Database Actions | SQL

Consumer group: LOW Data Load

Navigator Files ?

ADMIN Tables Search...

```

1 Select * from grooming_history;
2

```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

| | GROOMING_ID | GROOMER_NAME | GROOMING_DATE | GROOMING_TYPE |
|----|-------------|----------------------|------------------------|-------------------------|
| 1 | 1 | Mary Johnson | 5/10/2023, 12:00:00 A | Bath and Brush |
| 2 | 2 | David Wilson | 6/15/2023, 12:00:00 A | Haircut |
| 3 | 3 | Jessica Thompson | 7/20/2023, 12:00:00 A | Nail Trim |
| 4 | 4 | Christopher Martinez | 8/25/2023, 12:00:00 A | Ear Cleaning |
| 5 | 5 | Jennifer Davis | 9/30/2023, 12:00:00 A | Full Groom |
| 6 | 6 | Ryan Garcia | 10/5/2023, 12:00:00 A | De-shedding Treatment |
| 7 | 7 | Emma Hernandez | 11/10/2023, 12:00:00 A | Teeth Cleaning |
| 8 | 8 | Matthew Taylor | 12/15/2023, 12:00:00 A | Flea and Tick Treatment |
| 9 | 9 | Olivia Anderson | 1/20/2024, 12:00:00 A | Paw Trim |
| 10 | 10 | Sophia Brown | 2/25/2024, 12:00:00 A | Sanitary Trim |

1 0 0 0 | 12:50:50 AM - 10 rows total

27°C Mostly cloudy

Search

7:50 PM 4/17/2024

Creating a table named **employees_staff_details**

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top navigation bar, the URL is https://g769cec454fd27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet. The main area displays the following SQL code:

```
1 CREATE TABLE employees_staff_details (
2     emp_id INT PRIMARY KEY,
3     emp_name VARCHAR2(50) NOT NULL,
4     emp_contact VARCHAR2(13) NOT NULL CHECK (emp_contact NOT LIKE '%-%'),
5     emp_training VARCHAR2(50) NOT NULL,
6     emp_salary NUMERIC NOT NULL,
7     shelter_id INT REFERENCES shelter_branch(shelter_id),
8     manager_emp_id INT
9 );
10
```

Below the code, the message "Table EMPLOYEES_STAFF_DETAILS created." is displayed. The status bar at the bottom left shows "Elapsed: 00:00:00.025". The status bar at the bottom right shows "Powered by ORDS", the date "4/17/2024", and the time "8:49 PM".

Inserting values to employees_staff_details

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top navigation bar, the URL is https://g769cec454fd27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet. The main area displays the following SQL code:

```
1 INSERT INTO employees_staff_details VALUES (1, 'John Smith', '+1234567890', 'Management Training', 58000, 1, NULL);
2 INSERT INTO employees_staff_details VALUES (2, 'Emily Johnson', '+1987654321', 'Animal Handling Workshop', 45000, 2, 1);
3 INSERT INTO employees_staff_details VALUES (3, 'Michael Williams', '+1234567890', 'Veterinary Assistant Training', 55000, 1, 2);
4 INSERT INTO employees_staff_details VALUES (4, 'Emma Brown', '+1567896123', 'Grooming Certification Course', 48000, 2, 1);
5 INSERT INTO employees_staff_details VALUES (5, 'Daniel Jones', '+16543210987', 'Accounting Workshop', 60000, 1, 2);
6 INSERT INTO employees_staff_details VALUES (6, 'Olivia Davis', '+1223344556', 'Customer Relationship Management Training', 52000, 2, 1);
7 INSERT INTO employees_staff_details VALUES (7, 'Liam Miller', '+1345678901', 'Maintenance Training Program', 47000, 1, 2);
8 INSERT INTO employees_staff_details VALUES (8, 'Sophia Wilson', '+18986123456', 'Security Officer Training', 54000, 2, 1);
9 INSERT INTO employees_staff_details VALUES (9, 'Noah Taylor', '+1567890234', 'Marketing Strategy Workshop', 58000, 1, NULL);
10 INSERT INTO employees_staff_details VALUES (10, 'Ava Martinez', '+19765432890', 'IT Certification Program', 57000, 2, 1);
```

Below the code, three messages are displayed: "1 row inserted.", "Elapsed: 00:00:00.005", "1 row inserted.", "Elapsed: 00:00:00.004", and "1 row inserted.", "Elapsed: 00:00:00.004". The status bar at the bottom left shows "Elapsed: 00:00:00.005". The status bar at the bottom right shows "Powered by ORDS", the date "4/17/2024", and the time "8:49 PM".

The screenshot shows the Oracle Database Actions SQL worksheet interface. The query `Select * from employees_staff_details;` has been run, resulting in 10 rows of data. The columns are labeled: EMP_ID, EMP_NAME, EMP_CONTACT, EMP_TRAINING, EMP_SALARY, SHELTER_ID, and MANAGER_EMP_ID. The data includes various employee names, contact numbers, training details, salaries, and manager information.

| EMP_ID | EMP_NAME | EMP_CONTACT | EMP_TRAINING | EMP_SALARY | SHELTER_ID | MANAGER_EMP_ID |
|--------|------------------|-------------|-------------------------|------------|------------|----------------|
| 1 | John Smith | +1234567890 | Management Training | 50000 | 1 | (null) |
| 2 | Emily Johnson | +1987654321 | Animal Handling Wor | 45000 | 2 | 1 |
| 3 | Michael Williams | +1122334455 | Veterinary Assistant Ti | 55000 | 1 | 2 |
| 4 | Emma Brown | +1567890123 | Grooming Certificatio | 48000 | 2 | 1 |
| 5 | Daniel Jones | +1654321987 | Accounting Worksho | 60000 | 1 | 2 |
| 6 | Olivia Davis | +1223344556 | Customer Relationshi | 52000 | 2 | 1 |
| 7 | Liam Miller | +1345678901 | Maintenance Training | 47000 | 1 | 2 |
| 8 | Sophia Wilson | +1890123456 | Security Officer Traini | 54000 | 2 | 1 |
| 9 | Noah Taylor | +1567890234 | Marketing Strategy W | 58000 | 1 | (null) |
| 10 | Ava Martinez | +1765432890 | IT Certification Progr | 57000 | 2 | 1 |

Creating a table named employees_stage(New)

The screenshot shows the Oracle Database Actions SQL worksheet interface. A CREATE TABLE statement is being run:

```

1 CREATE TABLE employees_stage (
2   emp_education VARCHAR2(25) PRIMARY KEY,
3   emp_department VARCHAR2(25) NOT NULL,
4   emp_level VARCHAR2(2) NOT NULL
5 );
6

```

The table is created successfully, and the system outputs the following messages:

- Elapsed: 00:00:00.005
- 1 row inserted.
- Elapsed: 00:00:00.004
- 1 row inserted.
- Elapsed: 00:00:00.004
- Table EMPLOYEES_STAGE created.
- Elapsed: 00:00:00.018

Inserting values to employees_stage(New)

The screenshot shows the Oracle Database Actions SQL worksheet interface. The code entered is:

```
1 INSERT INTO employees_stage VALUES ('Bachelor in Administration', 'Administration', 'A1');
2 INSERT INTO employees_stage VALUES ('Master in Animal Care', 'Animal Care', 'A2');
3 INSERT INTO employees_stage VALUES ('Bachelor in Veterinary', 'Veterinary Services', 'B1');
4 INSERT INTO employees_stage VALUES ('Master in Grooming', 'Grooming', 'B2');
5 INSERT INTO employees_stage VALUES ('Master in Finance', 'Finance', 'C1');
6 INSERT INTO employees_stage VALUES ('Bachelor in Customer', 'Customer Service', 'C2');
7 INSERT INTO employees_stage VALUES ('Bachelor in Facility', 'Facility Maintenance', 'D1');
8 INSERT INTO employees_stage VALUES ('Bachelor in Security', 'Security', 'D2');
9 INSERT INTO employees_stage VALUES ('Master in Marketing', 'Marketing', 'E1');
10 INSERT INTO employees_stage VALUES ('Bachelor in IT', 'IT', 'E2');
```

The results pane shows:

Query Result

| 1 | row inserted. | |
|---|-----------------------|--|
| | Elapsed: 00:00:00.007 | |

Script Output

| 1 | row inserted. |
|---|-----------------------|
| | Elapsed: 00:00:00.007 |

DBMS Output

Explain Plan

Autotrace

SQL History

The screenshot shows the Oracle Database Actions SQL worksheet interface. The code entered is:

```
1 Select * from employees_stage;
```

The results pane shows:

Query Result

| 1 | Select * from employees_stage; | |
|---|--------------------------------|-------------------------------|
| | Download | Execution time: 0.017 seconds |

EMP_EDUCATION EMP_DEPARTMENT EMP_LEVEL

| | EMP_EDUCATION | EMP_DEPARTMENT | EMP_LEVEL |
|----|------------------------|---------------------|-----------|
| 1 | Bachelor | Administration | A1 |
| 2 | Associate | Animal Care | A2 |
| 3 | Master | Finance | C1 |
| 4 | Master in Animal Care | Animal Care | A2 |
| 5 | Master in Grooming | Grooming | B2 |
| 6 | Master in Finance | Finance | C1 |
| 7 | Bachelor in Security | Security | D2 |
| 8 | Master in Marketing | Marketing | E1 |
| 9 | Bachelor in IT | IT | E2 |
| 10 | Bachelor in Veterinary | Veterinary Services | B1 |
| 11 | Bachelor in Customer | Customer Service | C2 |

Creating a table named animal_info

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```
1 CREATE TABLE animal_info (
2     animal_id INT PRIMARY KEY,
3     a_type VARCHAR(58) NOT NULL,
4     a_breed VARCHAR(150) NOT NULL,
5     a_dob DATE NOT NULL,
6     date_of_acquisition DATE NOT NULL,
7     a_vaccine_status VARCHAR(58) NOT NULL,
8     animal_features_id INT REFERENCES animal_features(animal_features_id)
9 );
10 
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

Table ANIMAL_INFO created.
Elapsed: 00:00:00.024

Powered by ORDS

25°C Partly cloudy 9:00 PM 4/17/2024

Inserting values to animal_info

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```
1 INSERT INTO animal_info VALUES (1, 'Dog', 'Labrador Retriever', DATE '2019-03-15', DATE '2023-05-01', 'Up to date', 1);
2 INSERT INTO animal_info VALUES (2, 'Cat', 'Siamese', DATE '2026-07-20', DATE '2023-08-10', 'Not vaccinated', 2);
3 INSERT INTO animal_info VALUES (3, 'Rabbit', 'Holland Lop', DATE '2021-01-05', DATE '2023-09-15', 'Partially vaccinated', 3);
4 INSERT INTO animal_info VALUES (4, 'Parrot', 'African Grey', DATE '2018-11-10', DATE '2023-10-30', 'Up to date', 4);
5 INSERT INTO animal_info VALUES (5, 'Hamster', 'Syrian', DATE '2022-04-25', DATE '2023-11-05', 'Not vaccinated', 5);
6 INSERT INTO animal_info VALUES (6, 'Guinea Pig', 'Abyssinian', DATE '2020-09-08', DATE '2023-12-15', 'Partially vaccinated', 6);
7 INSERT INTO animal_info VALUES (7, 'Turtle', 'Red-eared Slider', DATE '2019-06-30', DATE '2024-01-25', 'Up to date', 7);
8 INSERT INTO animal_info VALUES (8, 'Fish', 'Goldfish', DATE '2021-03-12', DATE '2024-02-28', 'Not vaccinated', 8);
9 INSERT INTO animal_info VALUES (9, 'Bird', 'Cockatiel', DATE '2020-08-10', DATE '2024-03-10', 'Partially vaccinated', 9);
10 INSERT INTO animal_info VALUES (10, 'Snake', 'Ball Python', DATE '2017-12-03', DATE '2024-04-15', 'Up to date', 10);
11 
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

Powered by ORDS

Light rain Tomorrow 9:00 PM 4/17/2024

The screenshot shows the Oracle Database Actions SQL worksheet interface. The query window contains the following SQL statement:

```
[Worksheet]*
1  Select * from animal_info;
```

The results pane displays a table with 10 rows of data:

| | ANIMAL_ID | A_TYPE | A_BREED | A_DOB | DATE_OF_ACQUISITI | A_VACCINE_STATUS | ANIMAL_FEATURES |
|----|-----------|------------|--------------------|-------------------------|-------------------------|----------------------|-----------------|
| 1 | | Dog | Labrador Retriever | 3/15/2019, 12:00:00 AM | 5/1/2023, 12:00:00 AM | Up to date | 1 |
| 2 | | Cat | Siamese | 7/20/2020, 12:00:00 AM | 8/10/2023, 12:00:00 AM | Not vaccinated | 2 |
| 3 | | Rabbit | Holland Lop | 1/5/2021, 12:00:00 AM | 9/15/2023, 12:00:00 AM | Partially vaccinated | 3 |
| 4 | | Parrot | African Grey | 11/10/2018, 12:00:00 AM | 10/20/2023, 12:00:00 AM | Up to date | 4 |
| 5 | | Hamster | Syrian | 4/25/2022, 12:00:00 AM | 11/5/2023, 12:00:00 AM | Not vaccinated | 5 |
| 6 | | Guinea Pig | Abyssinian | 9/8/2020, 12:00:00 AM | 12/15/2023, 12:00:00 AM | Partially vaccinated | 6 |
| 7 | | Turtle | Red-eared Slider | 6/30/2019, 12:00:00 AM | 1/25/2024, 12:00:00 AM | Up to date | 7 |
| 8 | | Fish | Goldfish | 3/12/2021, 12:00:00 AM | 2/20/2024, 12:00:00 AM | Not vaccinated | 8 |
| 9 | | Bird | Cockatiel | 8/18/2020, 12:00:00 AM | 3/10/2024, 12:00:00 AM | Partially vaccinated | 9 |
| 10 | | Snake | Ball Python | 12/3/2017, 12:00:00 AM | 4/15/2024, 12:00:00 AM | Up to date | 10 |

Execution time: 0.009 seconds

Creating a table named animal_breed_info(New)

The screenshot shows the Oracle Database Actions SQL worksheet interface. The query window contains the following SQL statement:

```
[Worksheet]*
1  CREATE TABLE animal_breed_info (
2      a_breed VARCHAR2(50) PRIMARY KEY,
3      a_type VARCHAR2(50) NOT NULL
4  );
```

The results pane displays the output of the CREATE TABLE statement:

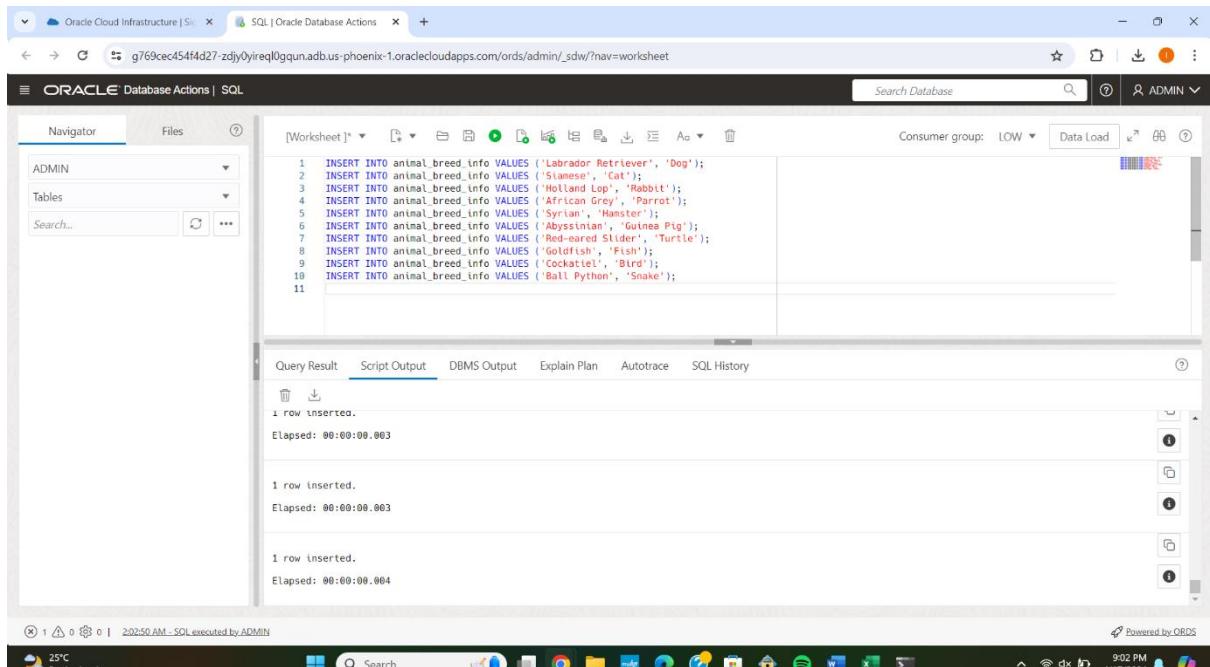
```
1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

Table ANIMAL_BREED_INFO created.
Elapsed: 00:00:00.016
```

Execution time: 0.0019 AM - SQL executed by ADMIN

Inserting values to animal_breed_info(New)



```

1  INSERT INTO animal_breed_info VALUES ('Labrador Retriever', 'Dog');
2  INSERT INTO animal_breed_info VALUES ('Siamese', 'Cat');
3  INSERT INTO animal_breed_info VALUES ('Holland Lop', 'Rabbit');
4  INSERT INTO animal_breed_info VALUES ('African Grey', 'Parrot');
5  INSERT INTO animal_breed_info VALUES ('Syrian', 'Hamster');
6  INSERT INTO animal_breed_info VALUES ('Abyssinian', 'Guinea Pig');
7  INSERT INTO animal_breed_info VALUES ('Red-eared Slider', 'Turtle');
8  INSERT INTO animal_breed_info VALUES ('Goldfish', 'Fish');
9  INSERT INTO animal_breed_info VALUES ('Cockatiel', 'Bird');
10 INSERT INTO animal_breed_info VALUES ('Ball Python', 'Snake');
11

```

Query Result tab selected. Output shows:

```

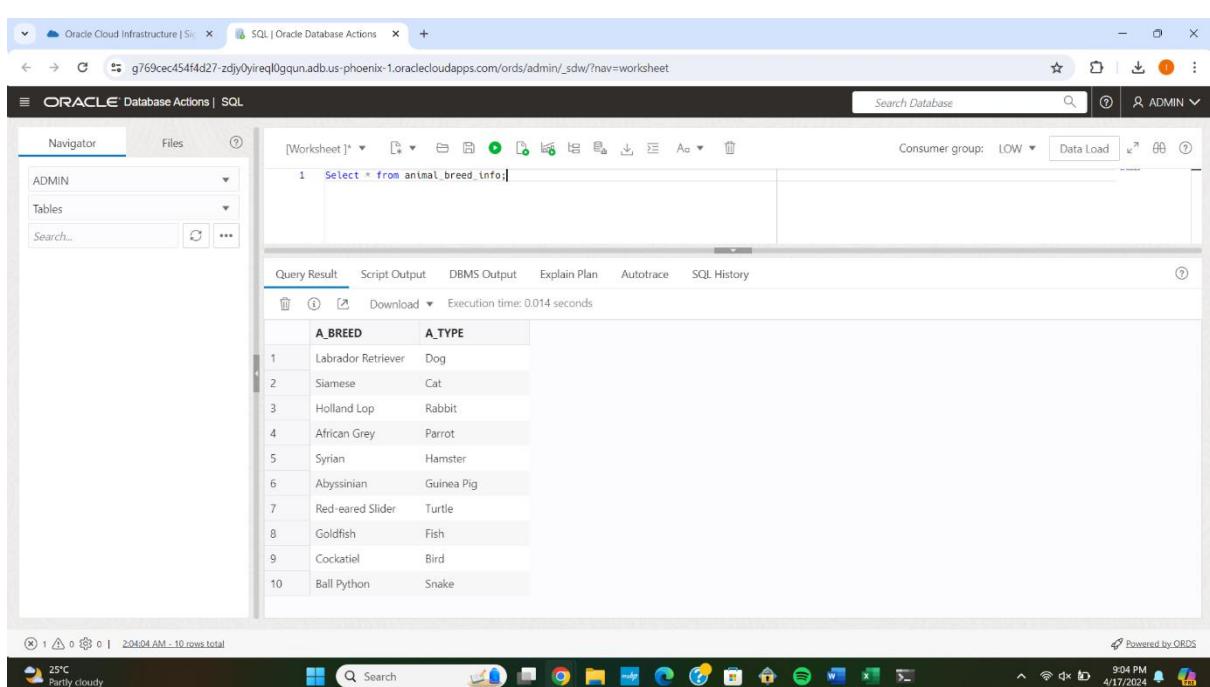
1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.004

```

Script Output tab selected. Shows the same three rows of output.



```

1  Select * from animal_breed_info;

```

Query Result tab selected. Output shows:

| A_BREED | A_TYPE |
|---------|--------------------|
| 1 | Labrador Retriever |
| 2 | Siamese |
| 3 | Holland Lop |
| 4 | African Grey |
| 5 | Syrian |
| 6 | Abyssinian |
| 7 | Red-eared Slider |
| 8 | Goldfish |
| 9 | Cockatiel |
| 10 | Ball Python |

Script Output tab selected. Shows the same results.

Creating a table named transfers

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left corner, there's a browser tab labeled "Oracle Cloud Infrastructure | SQL". Below it, the main window title is "SQL | Oracle Database Actions". The URL in the address bar is "g769cec454fd27-zdjy0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The interface includes a "Navigator" sidebar on the left with "ADMIN" selected, showing "Tables" and a "Search..." field. The main workspace contains the following SQL code:

```
1 CREATE TABLE transfers (
2     shelter_id INT REFERENCES shelter_branch(shelter_id),
3     transfer_id INT NOT NULL,
4     PRIMARY KEY (shelter_id, transfer_id)
5 );
6
```

Below the code, the "Script Output" tab is active, showing the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.005
```

Table TRANSFERS created.
Elapsed: 00:00:00.015

The status bar at the bottom indicates "2:10:36 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 9:10 PM".

Inserting values to transfers

This screenshot shows the same Oracle Database Actions SQL worksheet interface. The "Script Output" tab is active, displaying the results of an INSERT INTO statement:

```
1 INSERT INTO transfers VALUES (1, 1);
2 INSERT INTO transfers VALUES (2, 2);
3 INSERT INTO transfers VALUES (3, 3);
4 INSERT INTO transfers VALUES (4, 4);
5 INSERT INTO transfers VALUES (5, 5);
6 INSERT INTO transfers VALUES (6, 6);
7 INSERT INTO transfers VALUES (7, 7);
8 INSERT INTO transfers VALUES (8, 8);
9 INSERT INTO transfers VALUES (9, 9);
10 INSERT INTO transfers VALUES (10, 10);
11
```

The results show two sets of output, each indicating 1 row inserted and an elapsed time of 00:00:00.005.

The status bar at the bottom indicates "2:11:05 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 9:11 PM".

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The main area displays a SQL worksheet titled "[Worksheet]". The query entered is "Select * from transfers;". The results are shown in a table with two columns: "SHELTER_ID" and "TRANSFER_ID". The data consists of 10 rows, each with values (1, 1), (2, 2), (3, 3), (4, 4), (5, 5), (6, 6), (7, 7), (8, 8), (9, 9), and (10, 10). Below the table, it says "Execution time: 0.017 seconds". The bottom status bar shows "21:14 AM - 10 rows total" and "Powered by ORDS".

Creating a table named animal_transfer_records

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The main area displays a SQL worksheet titled "[Worksheet]". The query entered is a CREATE TABLE statement:

```
1 CREATE TABLE animal_transfer_records (
2     transfer_id INT PRIMARY KEY,
3     transfer_date DATE NOT NULL,
4     transfer_from VARCHAR2(25) NOT NULL,
5     transfer_to VARCHAR2(25) NOT NULL,
6     animal_id INT REFERENCES animal.info(animal_id)
7 );
8
```

The results show "Table ANIMAL_TRANSFER_RECORDS created." and "Elapsed: 00:00:00.023". The bottom status bar shows "21:23 AM - SQL executed by ADMIN" and "Powered by ORDS".

Inserting values to animal_transfer_records

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```

1  INSERT INTO animal_transfer_records VALUES (1, TO_DATE('2023-01-10', 'YYYY-MM-DD'), 'Shelter A', 'Shelter B', 1);
2  INSERT INTO animal_transfer_records VALUES (2, TO_DATE('2023-02-15', 'YYYY-MM-DD'), 'Shelter B', 'Shelter C', 2);
3  INSERT INTO animal_transfer_records VALUES (3, TO_DATE('2023-03-20', 'YYYY-MM-DD'), 'Shelter C', 'Shelter D', 3);
4  INSERT INTO animal_transfer_records VALUES (4, TO_DATE('2023-04-25', 'YYYY-MM-DD'), 'Shelter D', 'Shelter E', 4);
5  INSERT INTO animal_transfer_records VALUES (5, TO_DATE('2023-05-30', 'YYYY-MM-DD'), 'Shelter E', 'Shelter F', 5);
6  INSERT INTO animal_transfer_records VALUES (6, TO_DATE('2023-06-05', 'YYYY-MM-DD'), 'Shelter F', 'Shelter G', 6);
7  INSERT INTO animal_transfer_records VALUES (7, TO_DATE('2023-07-10', 'YYYY-MM-DD'), 'Shelter G', 'Shelter H', 7);
8  INSERT INTO animal_transfer_records VALUES (8, TO_DATE('2023-08-15', 'YYYY-MM-DD'), 'Shelter H', 'Shelter I', 8);
9  INSERT INTO animal_transfer_records VALUES (9, TO_DATE('2023-09-20', 'YYYY-MM-DD'), 'Shelter I', 'Shelter J', 9);
10 INSERT INTO animal_transfer_records VALUES (10, TO_DATE('2023-10-25', 'YYYY-MM-DD'), 'Shelter J', 'Shelter K', 10);
11

```

Elapsed: 00:00:00.006

1 row inserted.

Elapsed: 00:00:00.007

1 row inserted.

Elapsed: 00:00:00.007

Powered by ORDS

25°C Partly cloudy

9:13 PM 4/17/2024

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```
1  Select * from animal_transfer_records;
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

| TRANSFER_ID | TRANSFER_DATE | TRANSFER_FROM | TRANSFER_TO | ANIMAL_ID |
|-------------|----------------------|---------------|-------------|-----------|
| 1 | 1/10/2023, 12:00:00 | A Shelter A | Shelter B | 1 |
| 2 | 2/15/2023, 12:00:00 | A Shelter B | Shelter C | 2 |
| 3 | 3/20/2023, 12:00:00 | A Shelter C | Shelter D | 3 |
| 4 | 4/25/2023, 12:00:00 | A Shelter D | Shelter E | 4 |
| 5 | 5/30/2023, 12:00:00 | A Shelter E | Shelter F | 5 |
| 6 | 6/5/2023, 12:00:00 | A Shelter F | Shelter G | 6 |
| 7 | 7/10/2023, 12:00:00 | A Shelter G | Shelter H | 7 |
| 8 | 8/15/2023, 12:00:00 | A Shelter H | Shelter I | 8 |
| 9 | 9/20/2023, 12:00:00 | A Shelter I | Shelter J | 9 |
| 10 | 10/25/2023, 12:00:00 | Shelter J | Shelter K | 10 |

Powered by ORDS

25°C Partly cloudy

9:13 PM 4/17/2024

Creating a table named `employee_payroll`

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left corner, there's a browser tab labeled "Oracle Cloud Infrastructure | Six" and another tab labeled "SQL | Oracle Database Actions". The main area is titled "[Worksheet]". On the left sidebar, under "Tables", there is a table named "ADMIN". The central workspace contains the following SQL code:

```
1 CREATE TABLE employee_payroll (
2     swipe_id VARCHAR2(10) PRIMARY KEY,
3     swipe_date DATE NOT NULL,
4     check_in_time TIMESTAMP NOT NULL,
5     check_out_time TIMESTAMP NOT NULL,
6     emp_id INT REFERENCES employees.staff_details(emp_id)
7 );
```

Below the code, the "Script Output" tab is selected, showing the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.007
```

Table EMPLOYEE_PAYROLL created.
Elapsed: 00:00:00.015

The status bar at the bottom indicates "2:15:52 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 9:15 PM".

Inserting values to employee_payroll

The screenshot shows the Oracle Database Actions SQL worksheet interface. The layout is identical to the previous one, with the "SQL | Oracle Database Actions" tab selected. The "Script Output" tab is selected, showing the results of the execution:

```
1 INSERT INTO employee_payroll VALUES ('SW1081', TO_DATE('2023-01-01', 'YYYY-MM-DD'), TIMESTAMP '2023-01-01 00:00:00', TIMESTAMP '2023-01-01 17:00:00', '2023-01-01 17:00:00');
2 INSERT INTO employee_payroll VALUES ('SW1082', TO_DATE('2023-01-01', 'YYYY-MM-DD'), TIMESTAMP '2023-01-02 00:00:00', TIMESTAMP '2023-01-02 08:30:00', '2023-01-02 08:30:00');
3 INSERT INTO employee_payroll VALUES ('SW1083', TO_DATE('2023-01-03', 'YYYY-MM-DD'), TIMESTAMP '2023-01-03 00:00:00', TIMESTAMP '2023-01-03 16:00:00', '2023-01-03 16:00:00');
4 INSERT INTO employee_payroll VALUES ('SW1084', TO_DATE('2023-01-04', 'YYYY-MM-DD'), TIMESTAMP '2023-01-04 00:00:00', TIMESTAMP '2023-01-04 17:30:00', '2023-01-04 17:30:00');
5 INSERT INTO employee_payroll VALUES ('SW1085', TO_DATE('2023-01-05', 'YYYY-MM-DD'), TIMESTAMP '2023-01-05 00:00:00', TIMESTAMP '2023-01-05 16:45:00', '2023-01-05 16:45:00');
6 INSERT INTO employee_payroll VALUES ('SW1086', TO_DATE('2023-01-06', 'YYYY-MM-DD'), TIMESTAMP '2023-01-06 00:00:00', TIMESTAMP '2023-01-06 08:45:00', '2023-01-06 08:45:00');
7 INSERT INTO employee_payroll VALUES ('SW1087', TO_DATE('2023-01-07', 'YYYY-MM-DD'), TIMESTAMP '2023-01-07 00:00:00', TIMESTAMP '2023-01-07 17:30:00', '2023-01-07 17:30:00');
8 INSERT INTO employee_payroll VALUES ('SW1088', TO_DATE('2023-01-08', 'YYYY-MM-DD'), TIMESTAMP '2023-01-08 00:00:00', TIMESTAMP '2023-01-08 16:00:00', '2023-01-08 16:00:00');
9 INSERT INTO employee_payroll VALUES ('SW1089', TO_DATE('2023-01-09', 'YYYY-MM-DD'), TIMESTAMP '2023-01-09 00:00:00', TIMESTAMP '2023-01-09 17:00:00', '2023-01-09 17:00:00');
10 INSERT INTO employee_payroll VALUES ('SW1010', TO_DATE('2023-01-10', 'YYYY-MM-DD'), TIMESTAMP '2023-01-10 00:00:00', TIMESTAMP '2023-01-10 08:00:00', '2023-01-10 08:00:00');
```

Below the code, the "Script Output" tab is selected, showing the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.007
```

1 row inserted.
Elapsed: 00:00:00.006

The status bar at the bottom indicates "2:16:39 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 9:17 PM".

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The URL is "g769cec454f4d27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The main area is titled "[Worksheet]". A query is run: "Select * from employee_payroll;". The results are displayed in a table:

| | SWIPE_ID | SWIPE_DATE | CHECK_IN_TIME | CHECK_OUT_TIME | EMP_ID | |
|----|----------|------------|---------------|---------------------|---------------------|----|
| 1 | SW1001 | 1/1/2023 | 12:00:00 AM | 2023-01-01T08:00:00 | 2023-01-01T17:00:00 | 1 |
| 2 | SW1002 | 1/2/2023 | 12:00:00 AM | 2023-01-02T08:30:00 | 2023-01-02T16:30:00 | 2 |
| 3 | SW1003 | 1/3/2023 | 12:00:00 AM | 2023-01-03T09:00:00 | 2023-01-03T18:00:00 | 3 |
| 4 | SW1004 | 1/4/2023 | 12:00:00 AM | 2023-01-04T08:00:00 | 2023-01-04T17:30:00 | 4 |
| 5 | SW1005 | 1/5/2023 | 12:00:00 AM | 2023-01-05T08:15:00 | 2023-01-05T16:45:00 | 5 |
| 6 | SW1006 | 1/6/2023 | 12:00:00 AM | 2023-01-06T08:45:00 | 2023-01-06T17:15:00 | 6 |
| 7 | SW1007 | 1/7/2023 | 12:00:00 AM | 2023-01-07T08:30:00 | 2023-01-07T17:30:00 | 7 |
| 8 | SW1008 | 1/8/2023 | 12:00:00 AM | 2023-01-08T08:00:00 | 2023-01-08T16:00:00 | 8 |
| 9 | SW1009 | 1/9/2023 | 12:00:00 AM | 2023-01-09T09:00:00 | 2023-01-09T17:00:00 | 9 |
| 10 | SW1010 | 1/10/2023 | 12:00:00 AM | 2023-01-10T08:00:00 | 2023-01-10T16:30:00 | 10 |

At the bottom, there is a status bar showing "2:17:31 AM - 10 rows total", "Powered by ORDS", and a system tray with weather information (25°C Partly cloudy) and system icons.

Creating a table named employee_pay(New)

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The URL is "g769cec454f4d27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The main area is titled "[Worksheet]". A SQL script is run:

```
1 CREATE TABLE employee_pay (
2     emp_id INT PRIMARY KEY REFERENCES employees_staff_details(emp_id),
3     hourly_pay NUMERIC NOT NULL
4 );
5 
```

The results show the table was created successfully:

```
1 row inserted.  
Elapsed: 00:00:00.006  
  
1 row inserted.  
Elapsed: 00:00:00.007  
  
1 row inserted.  
Elapsed: 00:00:00.006  
  
Table EMPLOYEE_PAY created.  
Elapsed: 00:00:00.019
```

At the bottom, there is a status bar showing "2:19:47 AM - SQL executed by ADMIN", "Powered by ORDS", and a system tray with weather information (25°C Partly cloudy) and system icons.

Inserting values to employee_pay(New)

The screenshot shows the Oracle Database Actions interface with a SQL worksheet. The code entered is:

```
1 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (1, 28.50);
2 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (2, 18.75);
3 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (3, 22.00);
4 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (4, 19.25);
5 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (5, 21.00);
6 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (6, 28.00);
7 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (7, 23.50);
8 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (8, 19.75);
9 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (9, 21.50);
10 INSERT INTO employee_pay (emp_id, hourly_pay) VALUES (10, 20.25);
```

The results section shows:

Elapsed: 00:00:00.007
1 row inserted.
Elapsed: 00:00:00.006

The screenshot shows the Oracle Database Actions interface with a SQL worksheet. The code entered is:

```
1 Select * from employee_pay;
```

The results section shows a table with 10 rows:

| EMP_ID | HOURLY_PAY |
|--------|------------|
| 1 | 21 |
| 2 | 19 |
| 3 | 22 |
| 4 | 19 |
| 5 | 21 |
| 6 | 20 |
| 7 | 24 |
| 8 | 20 |
| 9 | 22 |
| 10 | 20 |

Elapsed: 00:00:00.001 seconds

Creating a table named support

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the code editor, the following SQL script is being run:

```
1 CREATE TABLE support (
2     shelter_id INT REFERENCES shelter.branch(shelter_id),
3     sponsor_id INT REFERENCES sponsor(sponsor_id),
4     PRIMARY KEY (shelter_id, sponsor_id)
5 );
6 
```

The results pane shows the execution log:

```
Elapsed: 00:00:00.007  
1 row inserted.  
Elapsed: 00:00:00.007  
1 row inserted.  
Elapsed: 00:00:00.006  
Table SUPPORT created.  
Elapsed: 00:00:00.021
```

The status bar at the bottom indicates "2:22:48 AM - SQL executed by ADMIN".

Inserting values to support

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the code editor, the following SQL script is being run:

```
1 INSERT INTO support VALUES (1, 1);
2 INSERT INTO support VALUES (2, 2);
3 INSERT INTO support VALUES (3, 3);
4 INSERT INTO support VALUES (4, 4);
5 INSERT INTO support VALUES (5, 5);
6 INSERT INTO support VALUES (6, 6);
7 INSERT INTO support VALUES (7, 7);
8 INSERT INTO support VALUES (8, 8);
9 INSERT INTO support VALUES (9, 9);
10 INSERT INTO support VALUES (10, 10);
11 
```

The results pane shows the execution log:

```
Elapsed: 00:00:00.006  
1 row inserted.  
Elapsed: 00:00:00.006  
1 row inserted.  
Elapsed: 00:00:00.007
```

The status bar at the bottom indicates "2:23:30 AM - SQL executed by ADMIN".

The screenshot shows the Oracle Database Actions interface. In the top-left, there's a 'Navigator' pane with 'ADMIN' selected. The main area contains a SQL worksheet with the following code:

```
1 Select * from support;
```

Below the code, the results are displayed in a table:

| | SHELTER_ID | SPONSOR_ID |
|----|------------|------------|
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
| 6 | 6 | 6 |
| 7 | 7 | 7 |
| 8 | 8 | 8 |
| 9 | 9 | 9 |
| 10 | 10 | 10 |

At the bottom of the worksheet, it says 'Execution time: 0.012 seconds'. The status bar at the bottom of the screen shows '2:24:02 AM - 10 rows total'.

Creating a table named accommodate

The screenshot shows the Oracle Database Actions interface. In the top-left, there's a 'Navigator' pane with 'ADMIN' selected. The main area contains a SQL worksheet with the following code:

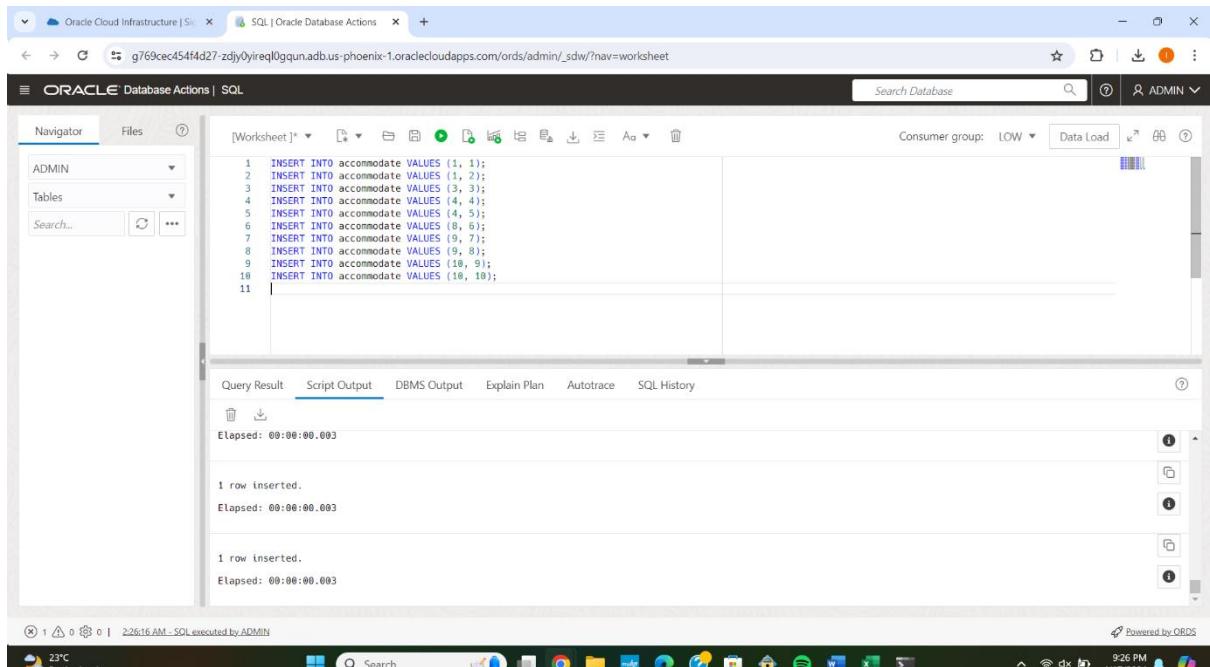
```
1 CREATE TABLE accommodate (
2     shelter_id INT REFERENCES shelter_branch(shelter_id),
3     animal_id INT NOT NULL,
4     PRIMARY KEY (shelter_id, animal_id)
5 );
```

Below the code, the results are displayed in a table:

| | SHELTER_ID | SPONSOR_ID |
|----|------------|------------|
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
| 6 | 6 | 6 |
| 7 | 7 | 7 |
| 8 | 8 | 8 |
| 9 | 9 | 9 |
| 10 | 10 | 10 |

At the bottom of the worksheet, it says 'Table ACCOMMODATE created.' and 'Elapsed: 00:00:00.021'. The status bar at the bottom of the screen shows '2:25:40 AM - SQL executed by ADMIN'.

Inserting values to accommodate



```

1 INSERT INTO accommodate VALUES (1, 1);
2 INSERT INTO accommodate VALUES (1, 2);
3 INSERT INTO accommodate VALUES (3, 3);
4 INSERT INTO accommodate VALUES (4, 4);
5 INSERT INTO accommodate VALUES (4, 5);
6 INSERT INTO accommodate VALUES (8, 6);
7 INSERT INTO accommodate VALUES (9, 7);
8 INSERT INTO accommodate VALUES (9, 8);
9 INSERT INTO accommodate VALUES (10, 9);
10 INSERT INTO accommodate VALUES (10, 10);
11

```

Elapsed: 00:00:00.003

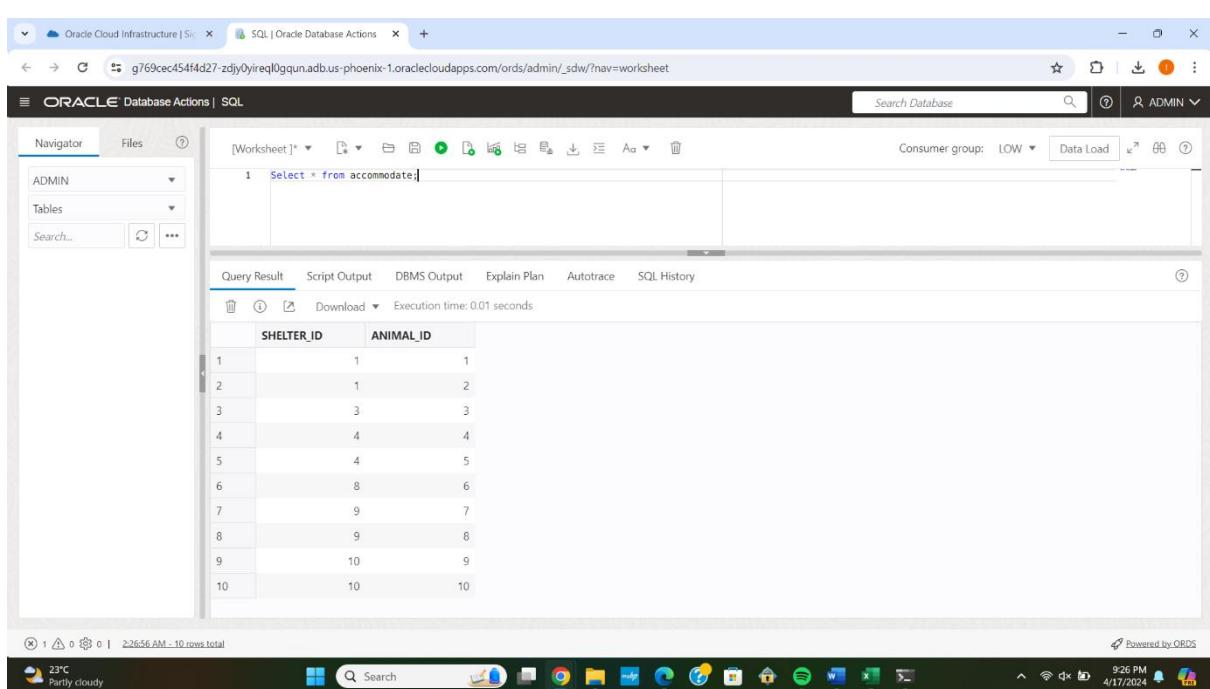
1 row inserted.

Elapsed: 00:00:00.003

1 row inserted.

Elapsed: 00:00:00.003

Powered by ORDS



```

1 Select * from accommodate;

```

| SHELTER_ID | ANIMAL_ID |
|------------|-----------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

Execution time: 0.01 seconds

Powered by ORDS

Creating a table named contain

The screenshot shows the Oracle Database Actions interface. In the top-left corner, there's a browser tab labeled "Oracle Cloud Infrastructure | SQL". Below it, the main window title is "SQL | Oracle Database Actions". The URL in the address bar is "g769cec454fd27-zdjy0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The interface includes a "Navigator" sidebar on the left with "ADMIN" selected, showing "Tables" and a "Search..." field. The main workspace is titled "[Worksheet]". It contains the following SQL code:

```
1 CREATE TABLE contain (
2     animal_id INT REFERENCES animal_info(animal_id),
3     ad_ID INT REFERENCES adoption_details(ad_ID),
4     PRIMARY KEY (animal_id, ad_ID)
5 );
```

Below the code, the "Query Result" tab is active, displaying the output of the executed query:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

Further down, the message "Table CONTAIN created." is shown, along with its execution time:

```
Elapsed: 00:00:00.029
```

The bottom status bar indicates the command was executed at "2:27:24 AM" by "ADMIN". The system tray shows the date and time as "4/17/2024 9:27 PM".

Inserting values to contain

This screenshot shows the same Oracle Database Actions interface as the previous one. The "Navigator" sidebar still has "ADMIN" selected. The main workspace contains the following SQL code:

```
1 INSERT INTO contain VALUES (1, 1);
2 INSERT INTO contain VALUES (2, 2);
3 INSERT INTO contain VALUES (3, 3);
4 INSERT INTO contain VALUES (4, 4);
5 INSERT INTO contain VALUES (5, 5);
6 INSERT INTO contain VALUES (6, 6);
7 INSERT INTO contain VALUES (7, 7);
8 INSERT INTO contain VALUES (8, 8);
9 INSERT INTO contain VALUES (9, 9);
10 INSERT INTO contain VALUES (10, 10);
```

The "Query Result" tab shows the results of the insertions:

```
Elapsed: 00:00:00.006
```

Three separate messages indicate successful insertions:

```
1 row inserted.  
Elapsed: 00:00:00.007
```

```
1 row inserted.  
Elapsed: 00:00:00.007
```

```
1 row inserted.  
Elapsed: 00:00:00.007
```

The bottom status bar shows the command was executed at "2:27:48 AM" by "ADMIN". The system tray shows the date and time as "4/17/2024 9:27 PM".

The screenshot shows the Oracle Database Actions interface. In the top-left corner, there's a browser tab for 'Oracle Cloud Infrastructure | SQL'. Below it, the main window title is 'SQL | Oracle Database Actions'. The URL is 'g769cec454fd27-zdjj0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet'. The interface includes a 'Navigator' sidebar with 'ADMIN' selected, showing 'Tables' and a 'Search...' field. The main workspace has a toolbar with various icons. A search bar at the top right says 'Search Database'. Below the toolbar, a message says 'Consumer group: LOW' and 'Data Load'. The central area contains a SQL worksheet with the following content:

```
[Worksheet]*
1 Select * from contain;
```

The 'Query Result' tab is active, showing the output of the query:

| ANIMAL_ID | AD_ID |
|-----------|-------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

Below the table, it says 'Execution time: 0.017 seconds'. The bottom status bar shows '2:26:38 AM - 10 rows total' and 'Powered by ORDS'. The system tray at the bottom indicates '23°C Partly cloudy'.

Creating a table named submit

This screenshot shows the Oracle Database Actions interface again. The browser tab and URL are identical to the previous screenshot. The main workspace shows a SQL worksheet with the following content:

```
1 CREATE TABLE submit (
2     s_id INT REFERENCES surrender_details(s_id),
3     animal_id INT REFERENCES animal_info(animal_id),
4     PRIMARY KEY (s_id, animal_id)
5 );
6
```

The 'Script Output' tab is active, showing the results of the execution:

Table SUBMIT created.
Elapsed: 00:00:00.019

The bottom status bar shows '2:29:06 AM - SQL executed by ADMIN' and 'Powered by ORDS'. The system tray at the bottom indicates '23°C Partly cloudy'.

Inserting values to submit

The screenshot displays two separate sessions in the Oracle Database Actions interface.

Session 1 (Top):

- Script:**

```

1  INSERT INTO submit VALUES (1, 1);
2  INSERT INTO submit VALUES (2, 2);
3  INSERT INTO submit VALUES (3, 3);
4  INSERT INTO submit VALUES (4, 4);
5  INSERT INTO submit VALUES (5, 5);
6  INSERT INTO submit VALUES (6, 6);
7  INSERT INTO submit VALUES (7, 7);
8  INSERT INTO submit VALUES (8, 8);
9  INSERT INTO submit VALUES (9, 9);
10 INSERT INTO submit VALUES (10, 10);

```
- Execution Results:**

1 row inserted.
Elapsed: 00:00:00.003

1 row inserted.
Elapsed: 00:00:00.003

Session 2 (Bottom):

- Query:**

```
1 Select * from submit;
```
- Execution Results:**

| S_ID | ANIMAL_ID |
|------|-----------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

10 rows total

Creating a table named need

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left corner, there's a browser tab labeled "Oracle Cloud Infrastructure | SQL". Below it, the main window title is "SQL | Oracle Database Actions". The URL in the address bar is "g769cec454f4d27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The interface includes a "Navigator" sidebar on the left with "ADMIN" selected, showing "Tables" and a "Search..." field. The main workspace contains the following SQL code:

```
1 CREATE TABLE need (
2   food_id INT REFERENCES nutrition(food_id),
3   animal_id INT REFERENCES animal_info(animal_id),
4   PRIMARY KEY (food_id, animal_id)
5 );
6 
```

Below the code, the "Query Result" tab is active, displaying the output:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

Further down, it shows:

```
Table NEED created.  
Elapsed: 00:00:00.016
```

The status bar at the bottom indicates "2:30:25 AM - SQL executed by ADMIN". The system tray shows the date and time as "4/17/2024 9:30 PM".

Inserting values to need

This screenshot shows the same Oracle Database Actions SQL worksheet interface. The top-left browser tab is still "Oracle Cloud Infrastructure | SQL". The main title is "SQL | Oracle Database Actions". The URL is "g769cec454f4d27-zdjy0ireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The "Navigator" sidebar shows "ADMIN" and "Tables". The main workspace contains the following SQL code:

```
1 INSERT INTO need VALUES (1, 1);
2 INSERT INTO need VALUES (2, 2);
3 INSERT INTO need VALUES (3, 3);
4 INSERT INTO need VALUES (4, 4);
5 INSERT INTO need VALUES (5, 5);
6 INSERT INTO need VALUES (6, 6);
7 INSERT INTO need VALUES (7, 7);
8 INSERT INTO need VALUES (8, 8);
9 INSERT INTO need VALUES (9, 9);
10 INSERT INTO need VALUES (10, 10);
11 
```

The "Query Result" tab is active, showing the output:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

Another section shows:

```
1 row inserted.  
Elapsed: 00:00:00.004
```

The status bar at the bottom indicates "2:31:23 AM - Code execution finished". The system tray shows the date and time as "4/17/2024 9:31 PM".

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Database Actions' and 'SQL'. The main area is a 'Worksheet' containing the following SQL code:

```
1 Select * from need;
```

The results pane shows a table with two columns: FOOD_ID and ANIMAL_ID. The data is as follows:

| | FOOD_ID | ANIMAL_ID |
|----|---------|-----------|
| 1 | 1 | 1 |
| 2 | 2 | 2 |
| 3 | 3 | 3 |
| 4 | 4 | 4 |
| 5 | 5 | 5 |
| 6 | 6 | 6 |
| 7 | 7 | 7 |
| 8 | 8 | 8 |
| 9 | 9 | 9 |
| 10 | 10 | 10 |

At the bottom of the results pane, it says 'Execution time: 0.016 seconds'.

Creating a table named done_by

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Database Actions' and 'SQL'. The main area is a 'Worksheet' containing the following SQL code:

```
1 CREATE TABLE done_by (
2     training_ID INT REFERENCES training_history(training_ID),
3     emp_id INT REFERENCES employees.staff_details(emp_id),
4     PRIMARY KEY (training_ID, emp_id)
5 );
```

The results pane shows the message 'Table DONE_BY created.' and 'Elapsed: 00:00:00.019'.

Inserting values to done_by

The screenshot shows two separate sessions in the Oracle Database Actions SQL Worksheet. Both sessions are executing the same sequence of SQL statements:

```

1 INSERT INTO done_by VALUES (1, 1);
2 INSERT INTO done_by VALUES (2, 2);
3 INSERT INTO done_by VALUES (3, 3);
4 INSERT INTO done_by VALUES (4, 4);
5 INSERT INTO done_by VALUES (5, 5);
6 INSERT INTO done_by VALUES (6, 6);
7 INSERT INTO done_by VALUES (7, 7);
8 INSERT INTO done_by VALUES (8, 8);
9 INSERT INTO done_by VALUES (9, 9);
10 INSERT INTO done_by VALUES (10, 10);

```

Both sessions show an elapsed time of 0:00:00.005 and report 1 row inserted.

The screenshot shows a session in the Oracle Database Actions SQL Worksheet executing a SELECT query:

```
1 Select * from done_by;
```

The results are displayed in a table:

| TRAINING_ID | EMP_ID |
|-------------|--------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

The session reports 10 rows total and an execution time of 0.01 seconds.

Creating a table named perform

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, there are tabs for 'Oracle Cloud Infrastructure | SQL' and 'SQL | Oracle Database Actions'. The main area is titled '[Worksheet]'. On the left, the 'Navigator' pane shows 'ADMIN' selected under 'Tables'. The central workspace contains the following SQL code:

```
1 CREATE TABLE perform (
2     grooming_id INT REFERENCES grooming_history(grooming_id),
3     emp_id INT REFERENCES employees.staff_details(emp_id),
4     PRIMARY KEY (grooming_id, emp_id)
5 );
```

Below the code, the 'Script Output' tab is active, displaying the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.006
```

Table PERFORM created.
Elapsed: 00:00:00.019

The status bar at the bottom indicates the time as 2:34:20 AM and the user as 'executed by ADMIN'. The system tray shows a weather icon for 23°C and a date/time stamp of 4/17/2024 at 9:34 PM.

Inserting values to perform

The screenshot shows the Oracle Database Actions interface. The top navigation bar has tabs for 'Oracle Cloud Infrastructure | SQL' and 'SQL | Oracle Database Actions'. The main area is titled '[Worksheet]'. The 'Navigator' pane on the left shows 'ADMIN' selected under 'Tables'. The central workspace contains the following SQL code:

```
1 INSERT INTO perform VALUES (1, 1);
2 INSERT INTO perform VALUES (2, 2);
3 INSERT INTO perform VALUES (3, 3);
4 INSERT INTO perform VALUES (4, 4);
5 INSERT INTO perform VALUES (5, 5);
6 INSERT INTO perform VALUES (6, 6);
7 INSERT INTO perform VALUES (7, 7);
8 INSERT INTO perform VALUES (8, 8);
9 INSERT INTO perform VALUES (9, 9);
10 INSERT INTO perform VALUES (10, 10);
```

The 'Script Output' tab is active, showing the results of the execution:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

Another row was inserted with the same elapsed time:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

The status bar at the bottom indicates the time as 2:34:47 AM and the user as 'executed by ADMIN'. The system tray shows a weather icon for 23°C and a date/time stamp of 4/17/2024 at 9:34 PM.

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The main area displays a SQL worksheet with the following content:

```
1 Select * from perform;
```

The results pane shows a table with two columns: GROOMING_ID and EMP_ID. The data is as follows:

| GROOMING_ID | EMP_ID |
|-------------|--------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

At the bottom of the interface, there is a status bar showing "2:35:33 AM - 10 rows total" and a system tray with icons for weather (23°C), battery, signal, and date/time (9:35 PM 4/17/2024).

Creating a table named examine

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The main area displays a SQL worksheet with the following content:

```
1 CREATE TABLE examine (
2     vet_id INT REFERENCES veterinary(vet_id),
3     medical_id INT REFERENCES medical_history(medical_id),
4     PRIMARY KEY (vet_id, medical_id)
5 );
```

The results pane shows the output of the create table command:

```
1 row inserted.  
Elapsed: 00:00:00.004
```

Three more rows are shown below, each indicating a successful insertion:

```
1 row inserted.  
Elapsed: 00:00:00.003
```

```
1 row inserted.  
Elapsed: 00:00:00.003
```

```
Table EXAMINE created.  
Elapsed: 00:00:00.022
```

At the bottom of the interface, there is a status bar showing "2:36:00 AM - SQL executed by ADMIN" and a system tray with icons for weather (23°C), battery, signal, and date/time (9:36 PM 4/17/2024).

Inserting values to examine

```
1 INSERT INTO examine VALUES (1, 1);
2 INSERT INTO examine VALUES (2, 2);
3 INSERT INTO examine VALUES (3, 3);
4 INSERT INTO examine VALUES (4, 4);
5 INSERT INTO examine VALUES (5, 5);
6 INSERT INTO examine VALUES (6, 6);
7 INSERT INTO examine VALUES (7, 7);
8 INSERT INTO examine VALUES (8, 8);
9 INSERT INTO examine VALUES (9, 9);
10 INSERT INTO examine VALUES (10, 10);
```

1 row inserted.
Elapsed: 00:00:00.006

2:36:26 AM - SQL executed by ADMIN

Powered by ORDS

```
1 Select * from examine;
```

| VET_ID | MEDICAL_ID |
|--------|------------|
| 1 | 1 |
| 2 | 2 |
| 3 | 3 |
| 4 | 4 |
| 5 | 5 |
| 6 | 6 |
| 7 | 7 |
| 8 | 8 |
| 9 | 9 |
| 10 | 10 |

10 rows total

Creating a table named veterinary_specialization

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows the ADMIN schema with Tables selected. The main workspace displays the following SQL code:

```
1 CREATE TABLE veterinary_specialization (
2     vet_id INT REFERENCES veterinary(vet_id),
3     specialization VARCHAR2(50),
4     PRIMARY KEY (vet_id, specialization)
5 );
```

Below the code, the Query Result tab shows the execution results:

```
1 row inserted.  
Elapsed: 00:00:00.005
```

Three more rows of identical output follow, indicating successful insertions of three additional rows. The final message is:

```
Table VETERINARY_SPECIALIZATION created.  
Elapsed: 00:00:00.016
```

The status bar at the bottom left indicates "237:12 AM - SQL executed by ADMIN". The system tray at the bottom right shows the date and time as "4/17/2024 9:37 PM".

Inserting values to `veterinary_specialization`

The screenshot shows the Oracle Database Actions SQL worksheet interface. In the top-left pane, the Navigator shows the ADMIN schema with Tables selected. The main workspace displays the following SQL code:

```
1 INSERT INTO veterinary_specialization VALUES (1, 'Internal Medicine');
2 INSERT INTO veterinary_specialization VALUES (2, 'Surgery');
3 INSERT INTO veterinary_specialization VALUES (3, 'Dermatology');
4 INSERT INTO veterinary_specialization VALUES (4, 'Oncology');
5 INSERT INTO veterinary_specialization VALUES (5, 'Cardiology');
6 INSERT INTO veterinary_specialization VALUES (6, 'Neurology');
7 INSERT INTO veterinary_specialization VALUES (7, 'Radiology');
8 INSERT INTO veterinary_specialization VALUES (8, 'Emergency and Critical Care');
9 INSERT INTO veterinary_specialization VALUES (9, 'Anesthesiology');
10 INSERT INTO veterinary_specialization VALUES (10, 'Behavioral Medicine');
```

Below the code, the Query Result tab shows the execution results:

```
Elapsed: 00:00:00.007
```

Three more rows of identical output follow, indicating successful insertions of three additional rows. The final message is:

```
1 row inserted.  
Elapsed: 00:00:00.007
```

The status bar at the bottom left indicates "237:50 AM - SQL executed by ADMIN". The system tray at the bottom right shows the date and time as "4/17/2024 9:37 PM".

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The URL is "g769cec454fd27-zdjj0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The main area displays the results of a query:

```
1 Select * from veterinary_specialization;
```

| VET_ID | SPECIALIZATION |
|--------|----------------------|
| 1 | Internal Medicine |
| 2 | Surgery |
| 3 | Dermatology |
| 4 | Oncology |
| 5 | Cardiology |
| 6 | Neurology |
| 7 | Radiology |
| 8 | Emergency and Critic |
| 9 | Anesthesiology |
| 10 | Behavioral Medicine |

Below the table, it says "Execution time: 0.013 seconds". The status bar at the bottom left shows "2:38:36 AM - 10 rows total". The system tray at the bottom right shows the date and time as "9:38 PM 4/17/2024".

Creating a table named stores

The screenshot shows the Oracle Database Actions interface. In the top navigation bar, it says "Oracle Cloud Infrastructure | SQL" and "SQL | Oracle Database Actions". The URL is "g769cec454fd27-zdjj0yireql0gqun.adb.us-phoenix-1.oraclecloudapps.com/ords/admin/_sdw?nav=worksheet". The main area displays the SQL code for creating the 'stores' table:

```
1 CREATE TABLE stores (
2     animal_info_id INT REFERENCES animal_info(animal_id),
3     training_history_id INT REFERENCES training.history(training_ID),
4     grooming_history_id INT REFERENCES grooming.history(grooming_id),
5     medical_history_id INT REFERENCES medical_history(medical_id),
6     PRIMARY KEY (animal_info_id, training.history_id, grooming.history_id, medical_history_id)
7 );
8
```

The results section shows "1 row inserted." and "Elapsed: 00:00:00.007". It also says "Table STORES created." and "Elapsed: 00:00:00.024". The status bar at the bottom left shows "2:39:11 AM - SQL executed by ADMIN". The system tray at the bottom right shows the date and time as "9:39 PM 4/17/2024".

Inserting values to stores

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```
1 INSERT INTO stores VALUES (1, 1, 1, 1);
2 INSERT INTO stores VALUES (2, 2, 2, 2);
3 INSERT INTO stores VALUES (3, 3, 3, 3);
4 INSERT INTO stores VALUES (4, 4, 4, 4);
5 INSERT INTO stores VALUES (5, 5, 5, 5);
6 INSERT INTO stores VALUES (6, 6, 6, 6);
7 INSERT INTO stores VALUES (7, 7, 7, 7);
8 INSERT INTO stores VALUES (8, 8, 8, 8);
9 INSERT INTO stores VALUES (9, 9, 9, 9);
10 INSERT INTO stores VALUES (10, 10, 10, 10);
11
```

Consumer group: LOW Data Load

Search Database (Ctrl+K)

ADMIN

Navigator Files

Tables Search...

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

1 row inserted.
Elapsed: 00:00:00.007

1 row inserted.
Elapsed: 00:00:00.007

1 row inserted.
Elapsed: 00:00:00.007

Powered by ORDS

Oracle Cloud Infrastructure | SQL | Oracle Database Actions

```
1 Select * from stores;
```

Consumer group: LOW Data Load

Search Database

ADMIN

Navigator Files

Tables Search...

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History

| ANIMAL_INFO_ID | TRAINING_HISTORY | GROOMING_HISTOF | MEDICAL_HISTORY_ |
|----------------|------------------|-----------------|------------------|
| 1 | 1 | 1 | 1 |
| 2 | 2 | 2 | 2 |
| 3 | 3 | 3 | 3 |
| 4 | 4 | 4 | 4 |
| 5 | 5 | 5 | 5 |
| 6 | 6 | 6 | 6 |
| 7 | 7 | 7 | 7 |
| 8 | 8 | 8 | 8 |
| 9 | 9 | 9 | 9 |
| 10 | 10 | 10 | 10 |

Execution time: 0.01 seconds

Powered by ORDS

INDIVIDUAL CONTRIBUTION

Firstly we started with understanding the functional dependencies and created multiple functional dependencies using the domain knowledge we have. For this we split the total tables into parts and decided to work on 3 to 4 tables each. I started with understand the animal_info tables and the functional dependencies in the table. After understanding that the breed type is different for different animals, I created a functional depends for the same. This indicates that for each animal breed type belongs to a animal type only. For this I found the need to split the table into 2 and mentioned the same to the team. By introducing this FD, we understood the need for dividing the table into 2 and divided it according and also proved that the dependency preservance is maintained and the join is lossless if joined.

Next I analysed the surrender details table and saw that there cannot be another table created or a functional dependency created as the assumptions we have made as per the project will not be in sync with the table if it is splitter into two. The defined functional dependencies were satisfying BCNF and thus nothing we changed. For the employee staff details table, I observed that we can define functional dependencies between the education and the level, department of the employee. As the employee is given a level depending on the education he/she has and also the department aligns with the education. I added two functional dependencies and these were not inline with the BCNF. So, there was a need to create another table to preserve these values. After this we also checked for lossless join and the join was lossless, proving that the BCFN can be done.

Then, on the employee payroll, similar analysis was done and the functional dependencies were identified as the hourly pay of the employee can be determined using the emp_id. So we created a new table for this and checked for lossless join to satisfy BCNF. Apart from this I have also updated the assumptions according and made changes to the SQL code and passed it to the teammates for further execution. I have also analysed and identified the placed where the tables should be added and the schema should be updated. I have analysed the need for showing the proof and the method to follow in order to prove that the join is lossless. Further, I mentioned and shared my knowledge about the super key and lossless joins in order to maintain synchronization.