**Animal Adoption Service Database**

**Forward Engineering Code**

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL,ALLOW\_INVALID\_DATES';

-- -----------------------------------------------------

-- Schema Animal\_Adoption\_Service

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema Animal\_Adoption\_Service

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `Animal\_Adoption\_Service` DEFAULT CHARACTER SET utf8 ;

USE `Animal\_Adoption\_Service` ;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Animal`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Animal` (

`animalID` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(255) NULL,

`gender` CHAR(6) NULL,

`type` VARCHAR(255) NULL,

`breed` VARCHAR(255) NULL,

`available` CHAR(3) NULL,

`price` FLOAT NULL,

PRIMARY KEY (`animalID`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Staff`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Staff` (

`staff\_id` INT NOT NULL AUTO\_INCREMENT,

`full\_name` VARCHAR(255) NULL,

`position` VARCHAR(255) NULL,

`address` VARCHAR(255) NULL,

`email` VARCHAR(255) NULL,

`phone\_number` LONGTEXT NULL,

`active` INT NULL,

PRIMARY KEY (`staff\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`AnimalRegister`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`AnimalRegister` (

`registration\_id` INT NOT NULL AUTO\_INCREMENT,

`registration\_date` DATE NOT NULL,

`weight` FLOAT NOT NULL,

`Animal\_animalID` INT NOT NULL,

`Staff\_staff\_id` INT NOT NULL,

PRIMARY KEY (`registration\_id`, `Animal\_animalID`, `Staff\_staff\_id`),

INDEX `fk\_AnimalRegister\_Animal1\_idx` (`Animal\_animalID` ASC),

INDEX `fk\_AnimalRegister\_Staff1\_idx` (`Staff\_staff\_id` ASC),

CONSTRAINT `fk\_AnimalRegister\_Animal1`

FOREIGN KEY (`Animal\_animalID`)

REFERENCES `Animal\_Adoption\_Service`.`Animal` (`animalID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_AnimalRegister\_Staff1`

FOREIGN KEY (`Staff\_staff\_id`)

REFERENCES `Animal\_Adoption\_Service`.`Staff` (`staff\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Customers`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Customers` (

`customer\_id` INT NOT NULL AUTO\_INCREMENT,

`full\_name` VARCHAR(255) NULL,

`address` VARCHAR(255) NULL,

`phone\_number` LONGTEXT NULL,

`budget` INT NOT NULL,

PRIMARY KEY (`customer\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Orders`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Orders` (

`order\_id` INT NOT NULL AUTO\_INCREMENT,

`order\_date` DATE NULL,

`Customers\_customer\_id` INT NOT NULL,

PRIMARY KEY (`order\_id`, `Customers\_customer\_id`),

INDEX `fk\_Orders\_Customers1\_idx` (`Customers\_customer\_id` ASC),

CONSTRAINT `fk\_Orders\_Customers1`

FOREIGN KEY (`Customers\_customer\_id`)

REFERENCES `Animal\_Adoption\_Service`.`Customers` (`customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Sales`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Sales` (

`Orders\_order\_id` INT NOT NULL,

`Orders\_Customers\_customer\_id` INT NOT NULL,

`Animal\_animalID` INT NOT NULL,

PRIMARY KEY (`Orders\_order\_id`, `Orders\_Customers\_customer\_id`, `Animal\_animalID`),

INDEX `fk\_Sales\_Animal1\_idx` (`Animal\_animalID` ASC),

CONSTRAINT `fk\_Sales\_Orders1`

FOREIGN KEY (`Orders\_order\_id` , `Orders\_Customers\_customer\_id`)

REFERENCES `Animal\_Adoption\_Service`.`Orders` (`order\_id` , `Customers\_customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Sales\_Animal1`

FOREIGN KEY (`Animal\_animalID`)

REFERENCES `Animal\_Adoption\_Service`.`Animal` (`animalID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Animal\_Adoption\_Service`.`Medical Records`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Animal\_Adoption\_Service`.`Medical Records` (

`medical\_records\_id` INT NOT NULL AUTO\_INCREMENT,

`medical\_status` VARCHAR(255) NULL,

`current\_health\_description` VARCHAR(255) NULL,

`neutered` CHAR(3) NULL,

`Animal\_animalID` INT NOT NULL,

PRIMARY KEY (`medical\_records\_id`, `Animal\_animalID`),

INDEX `fk\_Medical Records\_Animal1\_idx` (`Animal\_animalID` ASC),

CONSTRAINT `fk\_Medical Records\_Animal1`

FOREIGN KEY (`Animal\_animalID`)

REFERENCES `Animal\_Adoption\_Service`.`Animal` (`animalID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

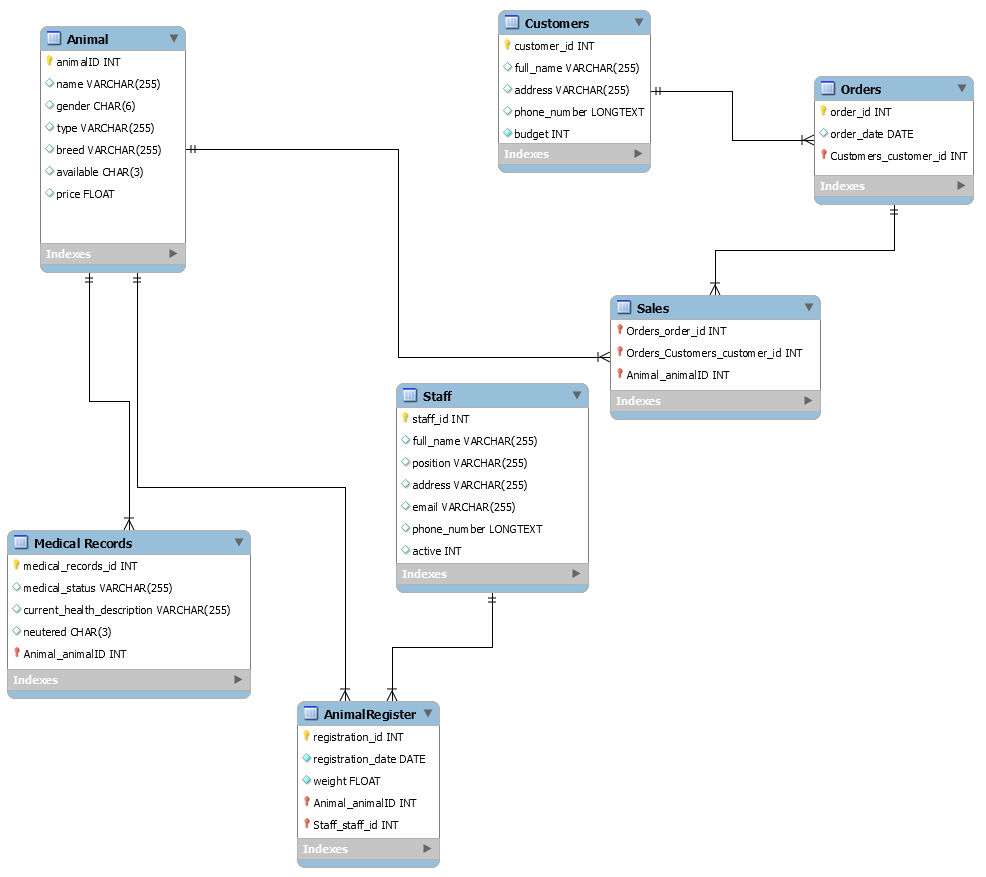
ENGINE = InnoDB;

SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

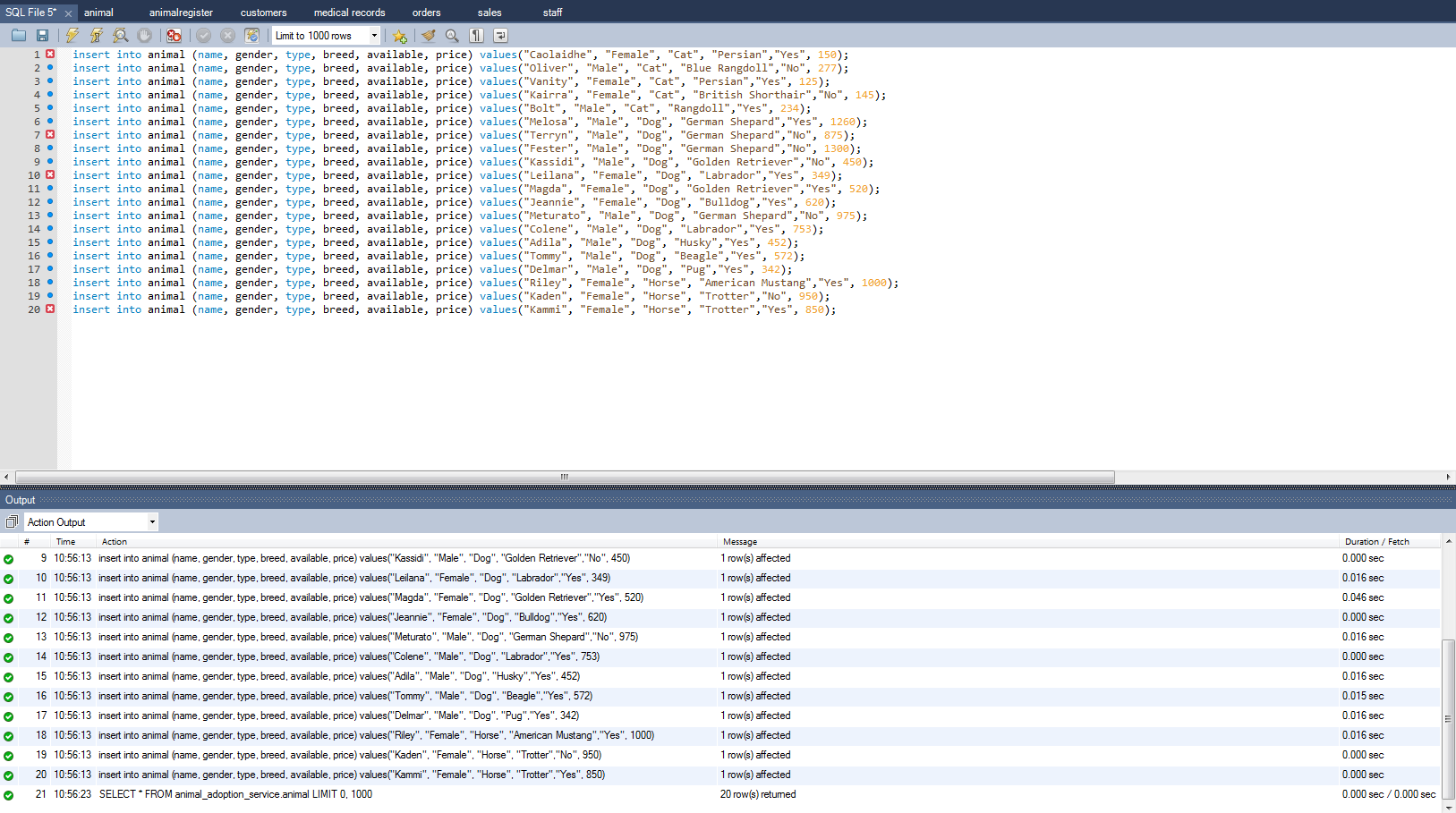
SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

**Animal Adoption Service ERD**

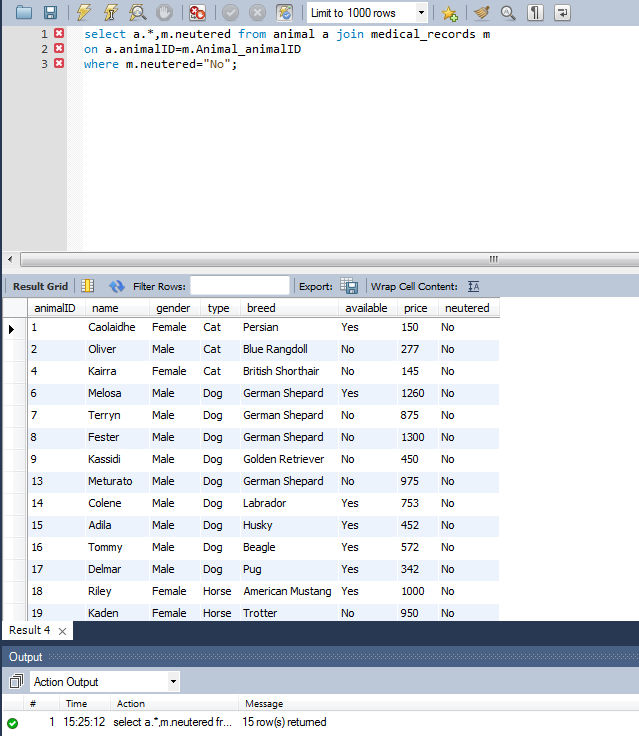
****

**User Stories:**

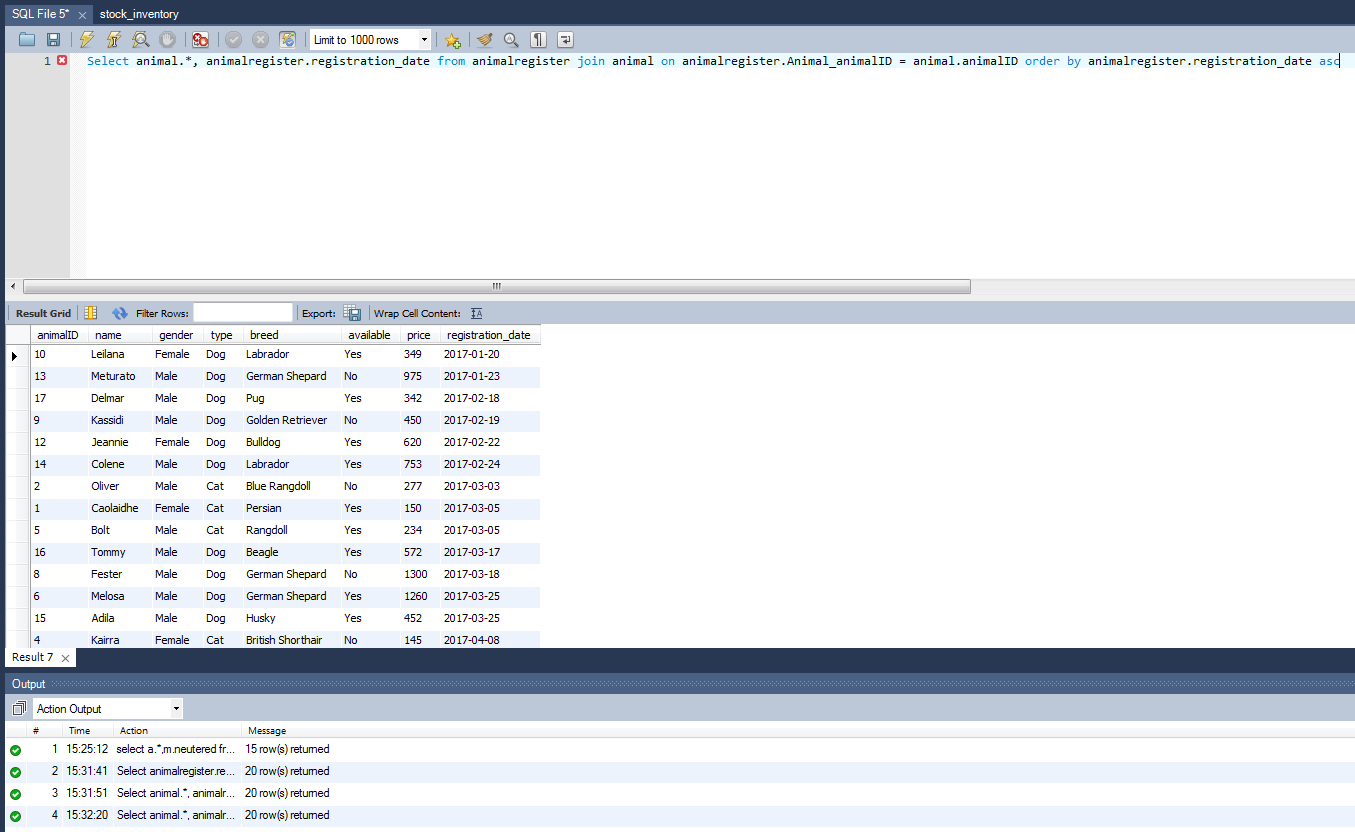
1. As a Health Care Officer, I want to query which animals have not been neutered so I can book an appointment with the vets
2. As a Social Media Consultant, I want to query which animals have been there the longest so I can create a campaign online to get them homed quickly
3. As a Receptionist, I want to query animals by specific criteria to help direct customers to the animals they wish to adopt
4. As a Sales Representative I want a query to generate sales report of the last 30 days in order to get rough idea about the average monthly selling and estimated gross income yearly
5. As a customer I want to query which animals are available for purchase between certain required age within the selected breed of the animal and between certain weight and within certain budget
6. As a Receptionist I want query to record the sale of the every animal and update their availability status to sold.
7. As a receptionist I want to return all the unique animal breeds with the total number of availability of animals in each category.
8. As a Sales Representative I want to return 3 most popular animals that are being bought by the customers frequently so I that I can maintain the stock level.
9. As a Sales Representative I want to return the list of the customers whom have signup with our company but haven’t bought the animal yet so that I can contact them and convince them to buy the animal
10. As a Sales Representative I want to return the top 3 customers whom have bought the animals frequently from us so that we can put them in lucky draw at the end of the year and the winner will get a voucher of our company as this is our way of thanking them for the support that they provide to these animals by adopting them.
11. As a medical expert I want to query the list of the animals which are ill or partially ill with the reason of their illness so that I can treat them in the correct manner.
12. As an Animal Trainer I want to query which animals are dogs and are part of the following breeds German Shepard, Golden Retriever, Labrador, Bulldog) so that I can give them the professional training and our sales representative might be able to sell them to the government agencies for the special purposes.

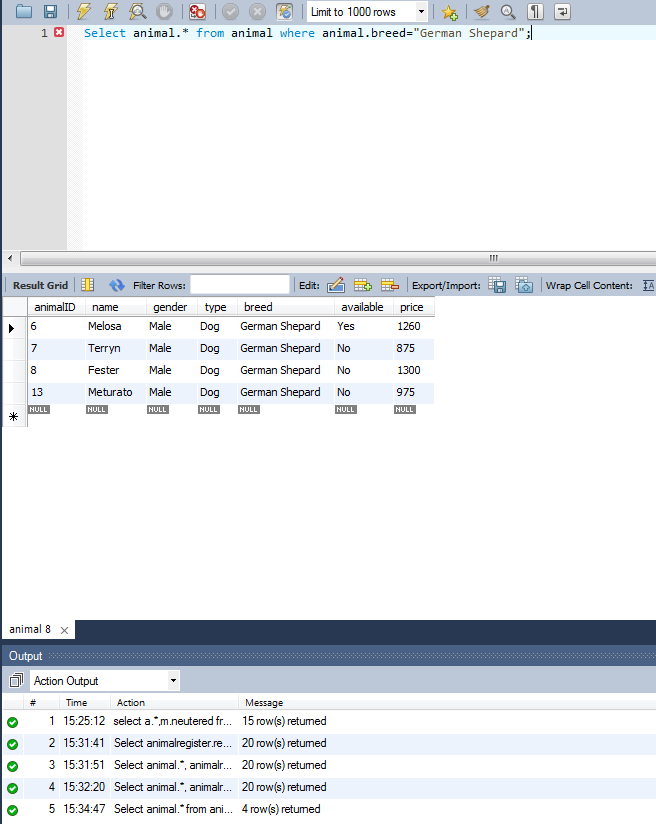
Insert Record

* As a Health Care Officer, I want to query which animals have not been neutered so I can book an appointment with the vets

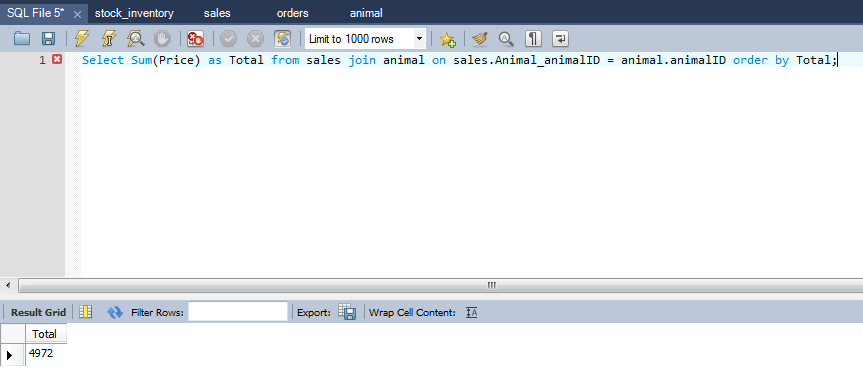


* As a Social Media Consultant, I want to query which animals have been there the longest so I can create a campaign online to get them homed quickly

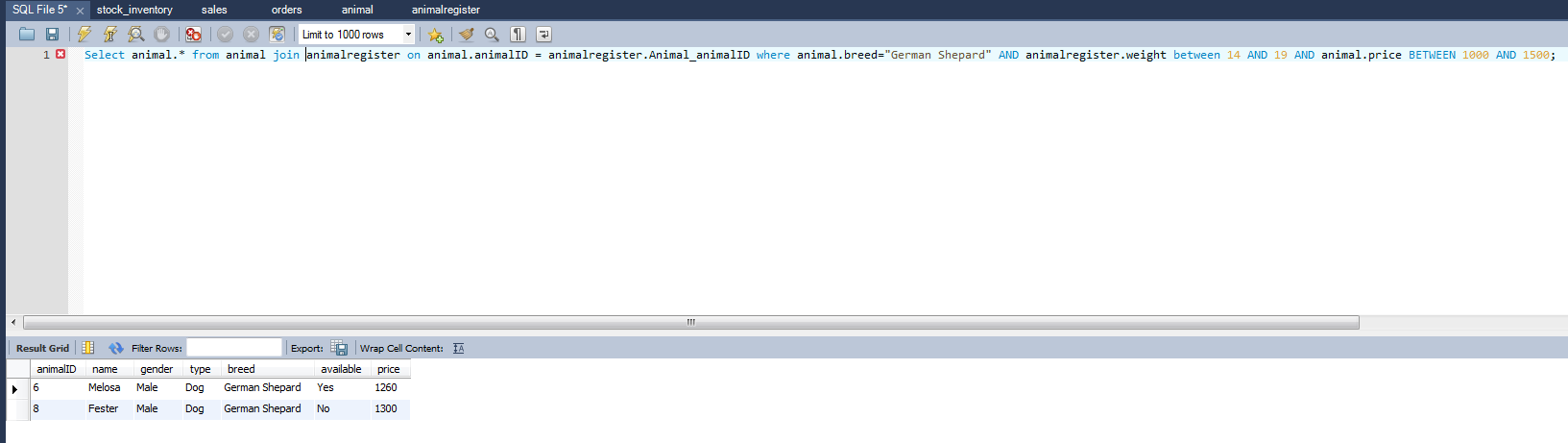


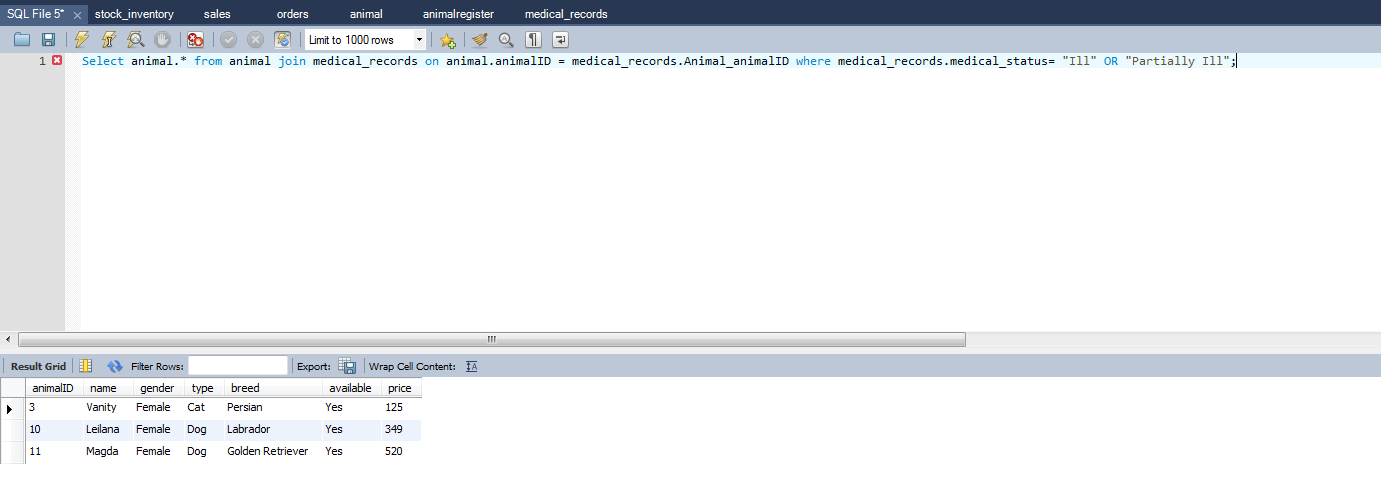
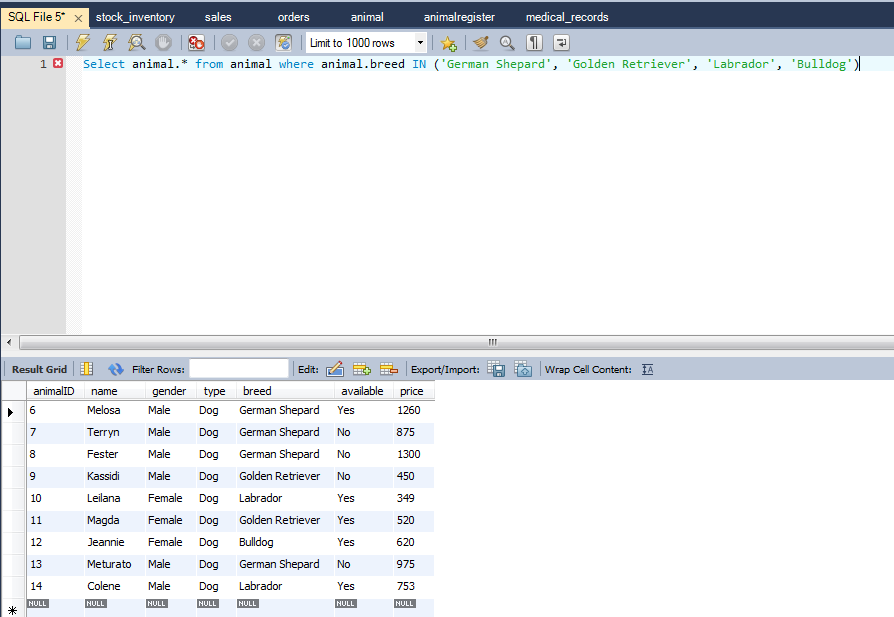
* As a Receptionist, I want to query animals by specific criteria to help direct customers to the animals they wish to adopt

* As a Sales Representative I want a query to generate sales report of the last 30 days in order to get rough idea about the average monthly selling and estimated gross income yearly

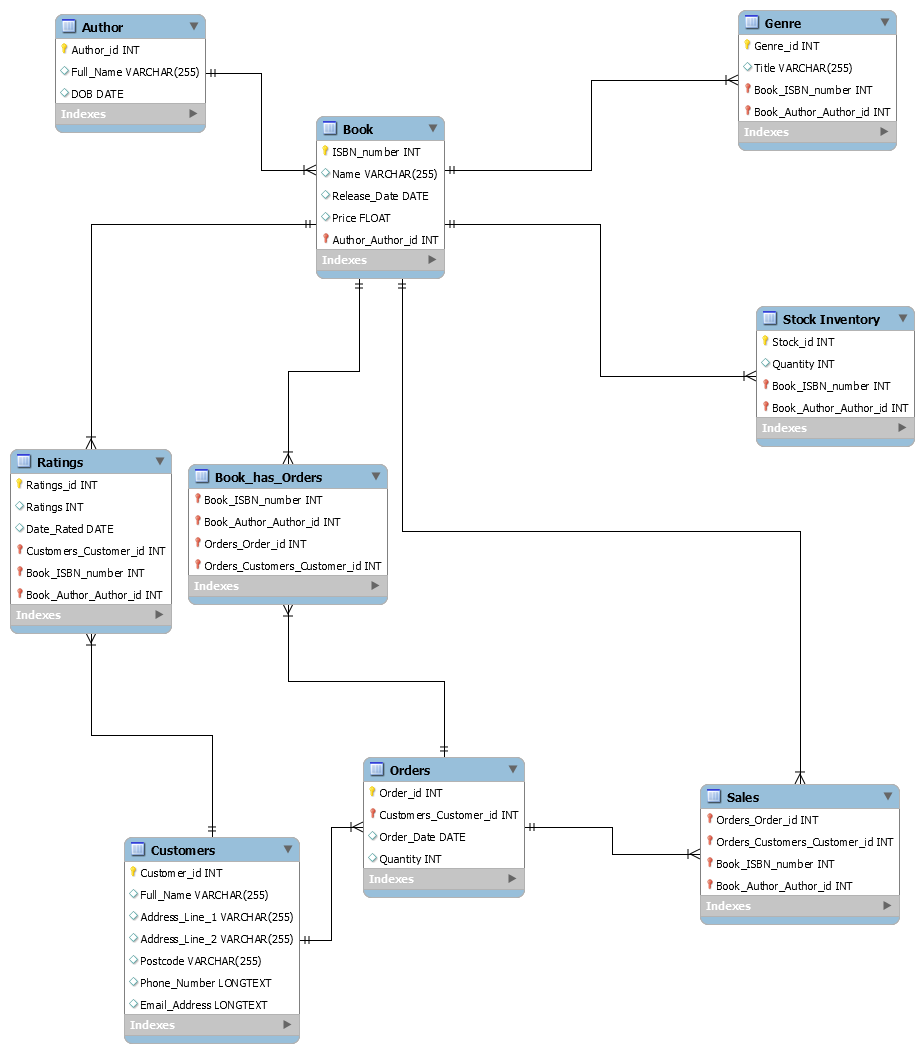


* As a customer I want to query which animals are available for purchase within the selected breed of the animal and between certain weight and within certain budget



* As a medical expert I want to query the list of the animals which are ill or partially ill with the reason of their illness so that I can treat them in the correct manner.
* As an Animal Trainer I want to query which animals are dogs and are part of the following breeds German Shepard, Golden Retriever, Labrador, Bulldog) so that I can give them the professional training and our sales representative might be able to sell them to the government agencies for the special purposes.

**Book Publishing ERD**

****

**Forward Engineering Code**

-- MySQL Workbench Forward Engineering

SET @OLD\_UNIQUE\_CHECKS=@@UNIQUE\_CHECKS, UNIQUE\_CHECKS=0;

SET @OLD\_FOREIGN\_KEY\_CHECKS=@@FOREIGN\_KEY\_CHECKS, FOREIGN\_KEY\_CHECKS=0;

SET @OLD\_SQL\_MODE=@@SQL\_MODE, SQL\_MODE='TRADITIONAL,ALLOW\_INVALID\_DATES';

-- -----------------------------------------------------

-- Schema Book\_Publication\_Company\_Database

-- -----------------------------------------------------

-- -----------------------------------------------------

-- Schema Book\_Publication\_Company\_Database

-- -----------------------------------------------------

CREATE SCHEMA IF NOT EXISTS `Book\_Publication\_Company\_Database` DEFAULT CHARACTER SET utf8 ;

USE `Book\_Publication\_Company\_Database` ;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Author`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Author` (

`Author\_id` INT NOT NULL AUTO\_INCREMENT,

`Full\_Name` VARCHAR(255) NULL,

`DOB` DATE NULL,

PRIMARY KEY (`Author\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Book`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Book` (

`ISBN\_number` INT NOT NULL,

`Name` VARCHAR(255) NULL,

`Release\_Date` DATE NULL,

`Price` FLOAT NULL,

`Author\_Author\_id` INT NOT NULL,

PRIMARY KEY (`ISBN\_number`, `Author\_Author\_id`),

INDEX `fk\_Book\_Author\_idx` (`Author\_Author\_id` ASC),

CONSTRAINT `fk\_Book\_Author`

FOREIGN KEY (`Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Author` (`Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Customers`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Customers` (

`Customer\_id` INT NOT NULL AUTO\_INCREMENT,

`Full\_Name` VARCHAR(255) NULL,

`Address\_Line\_1` VARCHAR(255) NULL,

`Address\_Line\_2` VARCHAR(255) NULL,

`Postcode` VARCHAR(255) NULL,

`Phone\_Number` LONGTEXT NULL,

`Email\_Address` LONGTEXT NULL,

PRIMARY KEY (`Customer\_id`))

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Orders`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Orders` (

`Order\_id` INT NOT NULL AUTO\_INCREMENT,

`Customers\_Customer\_id` INT NOT NULL,

`Order\_Date` DATE NULL,

`Quantity` INT NULL,

PRIMARY KEY (`Order\_id`, `Customers\_Customer\_id`),

INDEX `fk\_Orders\_Customers1\_idx` (`Customers\_Customer\_id` ASC),

CONSTRAINT `fk\_Orders\_Customers1`

FOREIGN KEY (`Customers\_Customer\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Customers` (`Customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Stock Inventory`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Stock Inventory` (

`Stock\_id` INT NOT NULL AUTO\_INCREMENT,

`Quantity` INT NULL,

`Book\_ISBN\_number` INT NOT NULL,

`Book\_Author\_Author\_id` INT NOT NULL,

PRIMARY KEY (`Stock\_id`, `Book\_ISBN\_number`, `Book\_Author\_Author\_id`),

INDEX `fk\_Stock Inventory\_Book1\_idx` (`Book\_ISBN\_number` ASC, `Book\_Author\_Author\_id` ASC),

CONSTRAINT `fk\_Stock Inventory\_Book1`

FOREIGN KEY (`Book\_ISBN\_number` , `Book\_Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Book` (`ISBN\_number` , `Author\_Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Ratings`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Ratings` (

`Ratings\_id` INT NOT NULL AUTO\_INCREMENT,

`Ratings` INT NULL,

`Date\_Rated` DATE NULL,

`Customers\_Customer\_id` INT NOT NULL,

`Book\_ISBN\_number` INT NOT NULL,

`Book\_Author\_Author\_id` INT NOT NULL,

PRIMARY KEY (`Ratings\_id`, `Customers\_Customer\_id`, `Book\_ISBN\_number`, `Book\_Author\_Author\_id`),

INDEX `fk\_Ratings\_Book1\_idx` (`Book\_ISBN\_number` ASC, `Book\_Author\_Author\_id` ASC),

CONSTRAINT `fk\_Ratings\_Customers1`

FOREIGN KEY (`Customers\_Customer\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Customers` (`Customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Ratings\_Book1`

FOREIGN KEY (`Book\_ISBN\_number` , `Book\_Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Book` (`ISBN\_number` , `Author\_Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Sales`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Sales` (

`Orders\_Order\_id` INT NOT NULL,

`Orders\_Customers\_Customer\_id` INT NOT NULL,

`Book\_ISBN\_number` INT NOT NULL,

`Book\_Author\_Author\_id` INT NOT NULL,

PRIMARY KEY (`Orders\_Order\_id`, `Orders\_Customers\_Customer\_id`, `Book\_ISBN\_number`, `Book\_Author\_Author\_id`),

INDEX `fk\_Sales\_Book1\_idx` (`Book\_ISBN\_number` ASC, `Book\_Author\_Author\_id` ASC),

CONSTRAINT `fk\_Sales\_Orders1`

FOREIGN KEY (`Orders\_Order\_id` , `Orders\_Customers\_Customer\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Orders` (`Order\_id` , `Customers\_Customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Sales\_Book1`

FOREIGN KEY (`Book\_ISBN\_number` , `Book\_Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Book` (`ISBN\_number` , `Author\_Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Genre`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Genre` (

`Genre\_id` INT NOT NULL AUTO\_INCREMENT,

`Title` VARCHAR(255) NULL,

`Book\_ISBN\_number` INT NOT NULL,

`Book\_Author\_Author\_id` INT NOT NULL,

PRIMARY KEY (`Genre\_id`, `Book\_ISBN\_number`, `Book\_Author\_Author\_id`),

CONSTRAINT `fk\_Genre\_Book1`

FOREIGN KEY (`Book\_ISBN\_number` , `Book\_Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Book` (`ISBN\_number` , `Author\_Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- -----------------------------------------------------

-- Table `Book\_Publication\_Company\_Database`.`Book\_has\_Orders`

-- -----------------------------------------------------

CREATE TABLE IF NOT EXISTS `Book\_Publication\_Company\_Database`.`Book\_has\_Orders` (

`Book\_ISBN\_number` INT NOT NULL,

`Book\_Author\_Author\_id` INT NOT NULL,

`Orders\_Order\_id` INT NOT NULL,

`Orders\_Customers\_Customer\_id` INT NOT NULL,

PRIMARY KEY (`Book\_ISBN\_number`, `Book\_Author\_Author\_id`, `Orders\_Order\_id`, `Orders\_Customers\_Customer\_id`),

INDEX `fk\_Book\_has\_Orders\_Orders1\_idx` (`Orders\_Order\_id` ASC, `Orders\_Customers\_Customer\_id` ASC),

INDEX `fk\_Book\_has\_Orders\_Book1\_idx` (`Book\_ISBN\_number` ASC, `Book\_Author\_Author\_id` ASC),

CONSTRAINT `fk\_Book\_has\_Orders\_Book1`

FOREIGN KEY (`Book\_ISBN\_number` , `Book\_Author\_Author\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Book` (`ISBN\_number` , `Author\_Author\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_Book\_has\_Orders\_Orders1`

FOREIGN KEY (`Orders\_Order\_id` , `Orders\_Customers\_Customer\_id`)

REFERENCES `Book\_Publication\_Company\_Database`.`Orders` (`Order\_id` , `Customers\_Customer\_id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

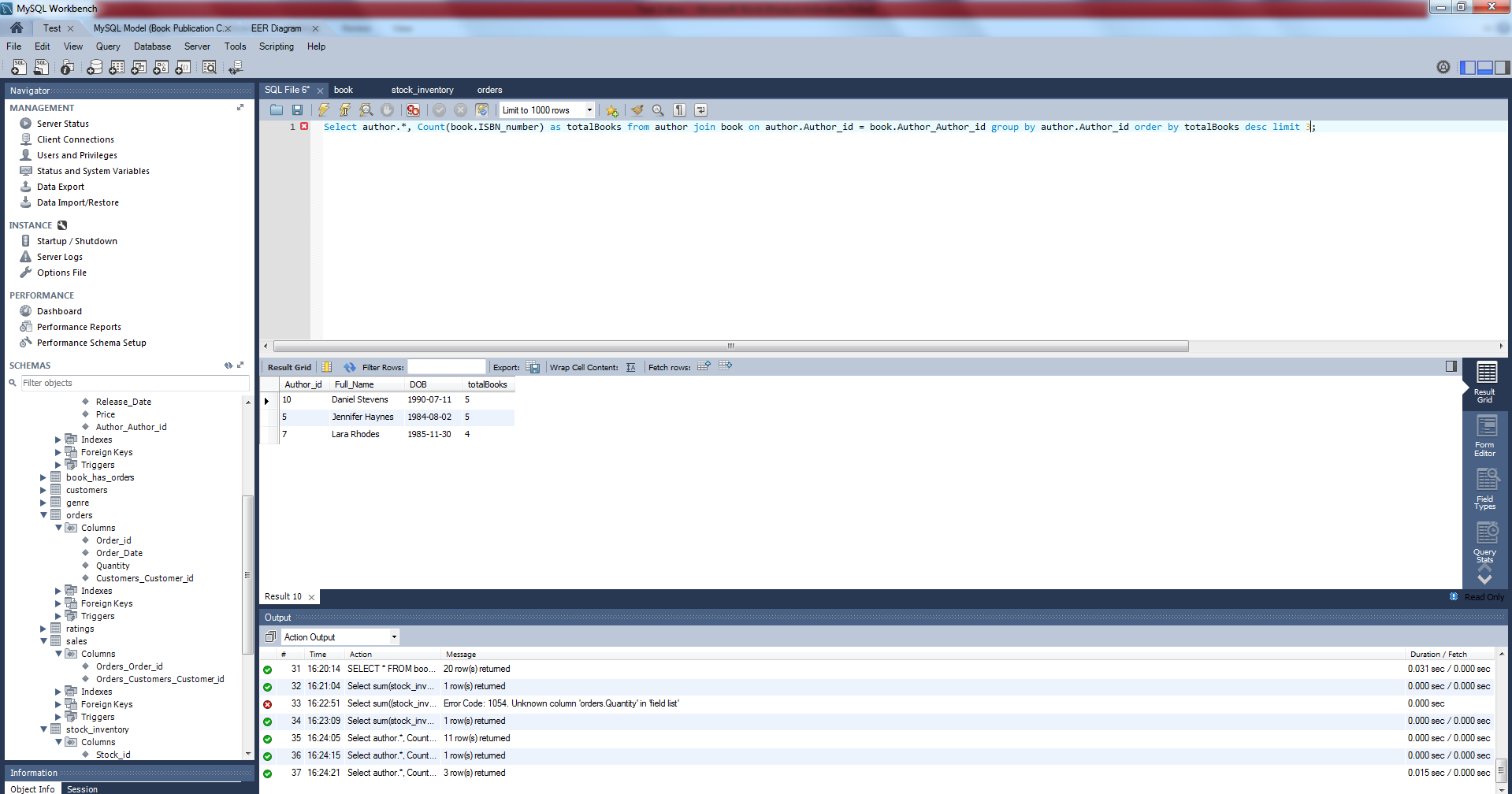
SET SQL\_MODE=@OLD\_SQL\_MODE;

SET FOREIGN\_KEY\_CHECKS=@OLD\_FOREIGN\_KEY\_CHECKS;

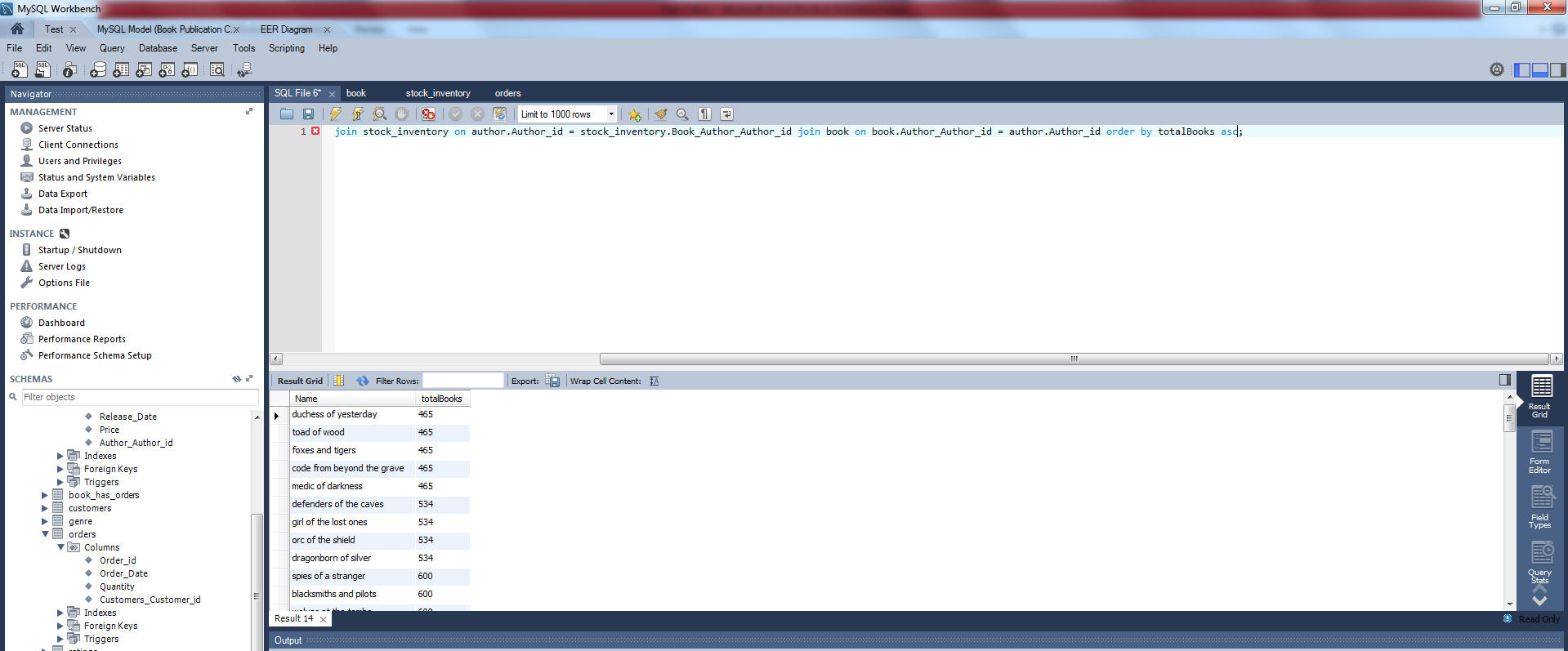
SET UNIQUE\_CHECKS=@OLD\_UNIQUE\_CHECKS;

**Book Publishing Company Database**

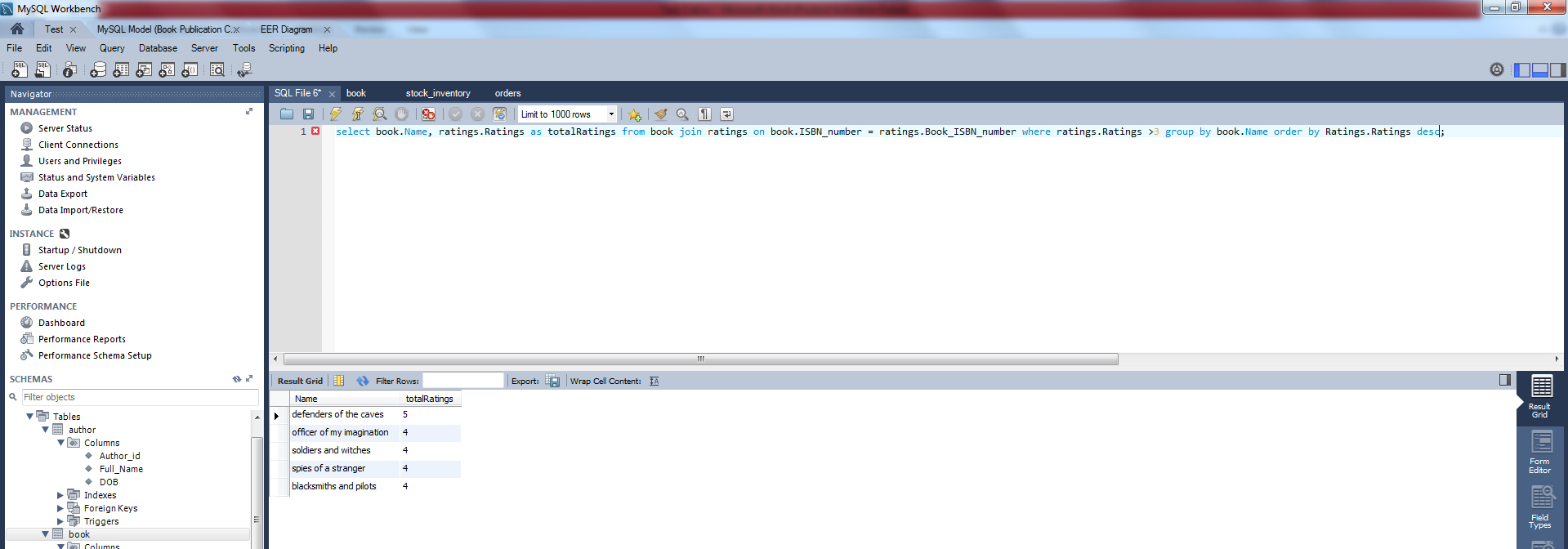
* As a Publishing Director, I want to search for our most lucrative author so I can appropriately reward them
* As a Marketing Executive, I wish to search for most popular genres by a specific author to create tailored advertising schemes
* As an Author Liaison, I want to search for books that are not yet complete by the order of the date they were started, so I can manage authors effectively
* As a Customer I want to query the list of the Action genre books that are being released between January 2017 to April 2017 and list them in the ascending order of high average customer review which helps me to decide which book I should buy
* As a Customer I want to query the most popular top 5 books which are being bought by the customers regularly.
* As a Sales Representative I want to query to check the stock availability for any books so we display the total number of books available on our website.
* As a Sales Representative I want to query the check the stock availability for the books that have stock level of less than 500 and even the total number of copies that are being sold so far since the release of the book so that I can decide whether to give reprinting order or to discontinue that particular book.
* As a Sales Representative I want a query to generate sales report of the last 30 days in order to get rough idea about the average monthly selling and estimated gross income yearly based on the total number of orders that are being received from the website and the store
* As a Marketing Representative I want a query of the customer whom have subscribed to our monthly newsletter so that I can send out the newsletters every month automatically.
* As a customer I want to query to retrieve all the books that have the ratings 3 or above.
* As a Publishing Director, I want to search for our most lucrative author so I can appropriately reward them



* As a Sales Representative I want to query to check the stock availability for any books so we display the total number of books available on our website.



* As a customer I want to query to retrieve all the books that have the ratings 3 or above.



**Task Code**

Task 2

1. create table person(personID int NOT NULL AUTO\_INCREMENT, fullName varchar(255), favouriteColour varchar(255) check(favouriteColour IN("blue", "red", "yellow", "green", "orange", "purple", "black", "white")), birthday date, favouriteNumber int check(birthday BETWEEN 1 AND 10), catchphrase varchar(255),PRIMARY KEY (personID));
2. create table pet(petID int NOT NULL AUTO\_INCREMENT, personID int,fullName varchar(255), name varchar(255), age int, gender char(6), type varchar(255) check(type IN("dog", "cat", "fish")), breed varchar(255) check(breed IN("shorthair", "bulldog", "betta")),PRIMARY KEY (petID),FOREIGN KEY (personID)references person(personID));

* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Steve Smith", "Blue", "1990-5-20", "122","you are the best");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Anna Clark", "Green", "1991-6-21", "22", "lazy person");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Roy Jackson", "Black", "1992-7-22", "34", "amazing person");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Gleen Rider", "Red", "1993-8-23", "45", "idiot person");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Steve Smith", "Navy", "1994-9-24", "54", "all the best");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Jack Johnson", "Dark Green", "1995-10-25", "654", "you can do it");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Amy WHS", "Light red", "1996-11-26", "225", "you got it right");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Alex KFC", "Sky Blue", "1997-12-27", "145", "nothing is impossible");
* insert into person (fullName, favouriteColour, birthday, favouriteNumber, catchphrase) values("Stuart Macdonald", "Maroon", "1998-1-28", "56" , "do it now");

1. delete from person where personid=1;
2. update person set favouriteNumber=501 where fullname="Anna Clark";
3. SELECT fullName, favouriteColour FROM person;

insert into pet (personID, name, age, gender, type, breed) values(7, "Rose", 5, "male", "dog","shorthair");

insert into pet (personID, name, age, gender, type, breed) values(3, "Jack", 4, "female", "cat", "bulldog");

insert into pet (personID, name, age, gender, type, breed) values(6, "Jackson", 11, "male", "monkey", "long hair");

insert into pet (personID, name, age, gender, type, breed) values(2, "Roaster", 13, "male", "dog", "shorthair");

insert into pet (personID, name, age, gender, type, breed) values(4, "Rider", 7, "female", "cat", "betta");

insert into pet (personID, name, age, gender, type, breed) values(6, "Roader", 8, "female", "rat", "bulldog");

insert into pet (personID, name, age, gender, type, breed) values(8, "Tougher", 15, "female", "rat", "bulldog");

insert into pet (personID, name, age, gender, type, breed) values(9, "Delight", 18, "female", "parrot" , "betta");

1. SELECT distinct breed from pet;
2. SELECT \* from person where favouriteNumber BETWEEN 3 AND 7;
3. SELECT \* from person where fullName like "c%";
4. SELECT fullName as newName, favouriteNumber+1 as newNo from person;
5. SELECT \* from person order by favouriteNumber DESC limit 5;
6. SELECT fullName, upper(substring(favouriteColour,1,3)) from person;
7. SELECT concat(fullName, " / ", catchphrase) from person;
8. SELECT avg(favouriteNumber) from person;
9. SELECT \*, max(age) as age from pet;
10. SELECT \*, min(age) as age from pet;
11. SELECT count(petID) as petID from pet
12. SELECT sum(petID) as petID from pet
13. SELECT avg(favouriteNumber) from person where favouriteColour='blue';
14. SELECT type, avg(age) as age from pet;
15. SELECT person.personID, pet.petID, person.fullName from person join pet ON person.personID = pet.petID;
16. SELECT person.personID, pet.petID, person.fullName from person left outer join pet ON person.personID = pet.personID where pet.petID is null;
17. SELECT personID, fullName, favouriteNumber from person UNION All SELECT petID, name, age FROM pet;
18. Create view dogViews as select \* from pet where type='dog';
19. SELECT \* from dogViews where age>5;

* Create user 'vetAdmin'@'localhost' identified by 'password';
* GRANT CREATE, UPDATE, SELECT ON information.pet TO 'vetAdmin'@'localhost';

Task 3

1. show tables;

* SELECT \* FROM information\_schema.columns WHERE table\_schema = 'sakila';
* describe actor;

1. select name from language order by name asc;
2. select first\_name, last\_name from actor where last\_name like "%MAN" order by first\_name asc;
3. select title, description from film where description REGEXP 'feminist|robot';
4. SELECT actor.first\_name, actor.last\_name, film.description from actor

INNER join film\_actor on actor.actor\_id = film\_actor.actor\_id

INNER join film on film.film\_id = film\_actor.film\_id where film.description REGEXP 'robot';

1. SELECT category.name, Count(film\_category.category\_id) AS results from category INNER join film\_category on category.category\_id = film\_category.category\_id INNER join film on film.film\_id = film\_category.film\_id GROUP BY film\_category.category\_id, category.name order by results DESC limit 5;
2. SELECT actor.first\_name, actor.last\_name, film.title FROM actor JOIN film\_actor ON actor.actor\_id = film\_actor.actor\_id JOIN film ON film.film\_id = film\_actor.film\_id;
3. SELECT customer.\*, Count(customer.customer\_id) as TotalOrders FROM customer JOIN payment ON customer.customer\_id = payment.customer\_id JOIN rental ON rental.rental\_id = payment.rental\_id group by customer.customer\_id order by TotalOrders desc limit 10;
4. SELECT customer.\*, Sum(payment.amount) as TotalOrders FROM customer JOIN payment ON customer.customer\_id = payment.customer\_id JOIN rental ON rental.rental\_id = payment.rental\_id group by customer.customer\_id order by TotalOrders asc limit 10;
5. SELECT distinct address.district from address;
6. SELECT Count(customer.customer\_id) as InactiveUsers from customer where customer.active=0;
7. SELECT store\_id, Count(store\_id) as PopularStore from customer group by store\_id order by PopularStore desc limit 1;
8. SELECT distinct rating from film;
9. select rating, count(film\_id) as TotalFilms from film group by rating;
10. select film.rating, count(film.film\_id) as TotalFilms from film JOIN inventory ON film.film\_id = inventory.film\_id group by rating;
11. SELECT film.title, count(rental.inventory\_id) as popular from inventory join rental ON inventory.inventory\_id = rental.inventory\_id join film on film.film\_id = inventory.film\_id group by film.film\_id order by popular desc;

Task 4

1. select \* from information\_schema.columns where table\_schema = 'movielens';

* select count(movies.id) as TotalMovies from movies;
* select count(users.id) as TotalUsers from users;
* select count(occupations.id) as TotalOccupations from occupations;
* select count(ratings.id) as TotalRatings from ratings;

1. select movies.title, count(movies.id) as TotalReviews from movies join ratings on movies.id = ratings.movie\_id group by movies.id order by TotalReviews desc;
2. select movies.title, count(movies.id) as TotalReviews, avg(ratings.rating) as AvgRatings from movies join ratings on movies.id = ratings.movie\_id group by movies.id order by AvgRatings desc;
3. Select genres.name, count(genres.id) as CommonGenre from genres join genres\_movies on genres.id = genres\_movies.genre\_id group by genres.id order by CommonGenre desc limit 1;