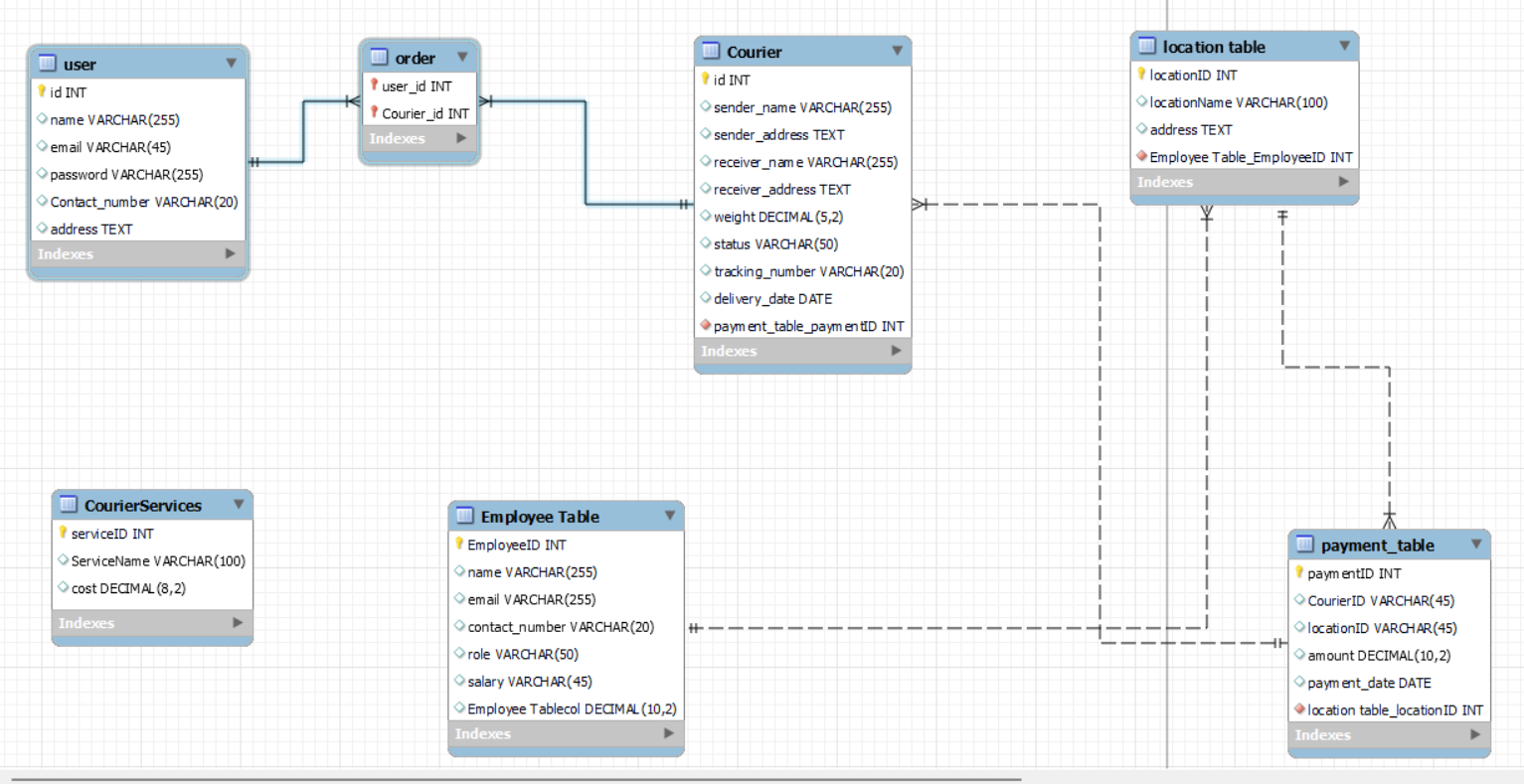
ASSIGNMENT 2 – COURIER MANAGEMENT



-- MySQL Workbench Forward Engineering

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

-- Schema courier\_management

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

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

-- Schema courier\_management

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

CREATE SCHEMA IF NOT EXISTS `courier\_management` DEFAULT CHARACTER SET utf8 ;

USE `courier\_management` ;

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

-- Table `courier\_management`.`user`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`user` (

`id` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(255) NULL,

`email` VARCHAR(45) NULL,

`password` VARCHAR(255) NULL,

`Contact\_number` VARCHAR(20) NULL,

`address` TEXT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `email\_UNIQUE` (`email` ASC) )

ENGINE = InnoDB;

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

-- Table `courier\_management`.`Employee Table`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`Employee Table` (

`EmployeeID` INT NOT NULL AUTO\_INCREMENT,

`name` VARCHAR(255) NULL,

`email` VARCHAR(255) NULL,

`contact\_number` VARCHAR(20) NULL,

`role` VARCHAR(50) NULL,

`salary` VARCHAR(45) NULL,

`Employee Tablecol` DECIMAL(10,2) NULL,

PRIMARY KEY (`EmployeeID`),

UNIQUE INDEX `email\_UNIQUE` (`email` ASC) )

ENGINE = InnoDB;

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

-- Table `courier\_management`.`location table`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`location table` (

`locationID` INT NOT NULL,

`locationName` VARCHAR(100) NULL,

`address` TEXT NULL,

`Employee Table\_EmployeeID` INT NOT NULL,

PRIMARY KEY (`locationID`),

INDEX `fk\_location table\_Employee Table\_idx` (`Employee Table\_EmployeeID` ASC) ,

CONSTRAINT `fk\_location table\_Employee Table`

FOREIGN KEY (`Employee Table\_EmployeeID`)

REFERENCES `courier\_management`.`Employee Table` (`EmployeeID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

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

-- Table `courier\_management`.`payment\_table`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`payment\_table` (

`paymentID` INT NOT NULL AUTO\_INCREMENT,

`CourierID` VARCHAR(45) NULL,

`locationID` VARCHAR(45) NULL,

`amount` DECIMAL(10,2) NULL,

`payment\_date` DATE NULL,

`location table\_locationID` INT NOT NULL,

PRIMARY KEY (`paymentID`),

INDEX `fk\_payment\_table\_location table1\_idx` (`location table\_locationID` ASC) ,

CONSTRAINT `fk\_payment\_table\_location table1`

FOREIGN KEY (`location table\_locationID`)

REFERENCES `courier\_management`.`location table` (`locationID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

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

-- Table `courier\_management`.`Courier`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`Courier` (

`id` INT NOT NULL AUTO\_INCREMENT,

`sender\_name` VARCHAR(255) NULL,

`sender\_address` TEXT NULL,

`receiver\_name` VARCHAR(255) NULL,

`receiver\_address` TEXT NULL,

`weight` DECIMAL(5,2) NULL,

`status` VARCHAR(50) NULL,

`tracking\_number` VARCHAR(20) NULL,

`delivery\_date` DATE NULL,

`payment\_table\_paymentID` INT NOT NULL,

PRIMARY KEY (`id`),

UNIQUE INDEX `tracking\_number\_UNIQUE` (`tracking\_number` ASC) ,

INDEX `fk\_Courier\_payment\_table1\_idx` (`payment\_table\_paymentID` ASC) ,

CONSTRAINT `fk\_Courier\_payment\_table1`

FOREIGN KEY (`payment\_table\_paymentID`)

REFERENCES `courier\_management`.`payment\_table` (`paymentID`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

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

-- Table `courier\_management`.`CourierServices`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`CourierServices` (

`serviceID` INT NOT NULL AUTO\_INCREMENT,

`ServiceName` VARCHAR(100) NULL,

`cost` DECIMAL(8,2) NULL,

PRIMARY KEY (`serviceID`))

ENGINE = InnoDB;

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

-- Table `courier\_management`.`order`

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

CREATE TABLE IF NOT EXISTS `courier\_management`.`order` (

`user\_id` INT NOT NULL,

`Courier\_id` INT NOT NULL,

PRIMARY KEY (`user\_id`, `Courier\_id`),

INDEX `fk\_user\_has\_Courier\_Courier1\_idx` (`Courier\_id` ASC) ,

INDEX `fk\_user\_has\_Courier\_user1\_idx` (`user\_id` ASC) ,

CONSTRAINT `fk\_user\_has\_Courier\_user1`

FOREIGN KEY (`user\_id`)

REFERENCES `courier\_management`.`user` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

CONSTRAINT `fk\_user\_has\_Courier\_Courier1`

FOREIGN KEY (`Courier\_id`)

REFERENCES `courier\_management`.`Courier` (`id`)

ON DELETE NO ACTION

ON UPDATE NO ACTION)

ENGINE = InnoDB;

-- Inserting sample data into the user\_table

INSERT INTO user\_table (name, email, password, contact\_num, address) VALUES

('Ajay', 'ajay@gmail.com', 'abc', '1234567890', '123 Main St'),

('Krishna', 'krishna@gmail.com', 'efg', '2345678901', '456 Elm St'),

('Vivek', 'vivek@gmail.com', 'hij', '3456789012', '789 Oak St'),

('Niranjan', 'niranjan@gmail.com', 'xyz', '4567890123', '101 Pine St'),

('Kowshik', 'kowshik@gmail.com', 'mno', '5678901234', '202 Maple St');

-- Inserting sample data into the courier table

INSERT INTO courier (SenderName, SenderAddress, ReceiverName, ReceiverAddress, weight, status, trackingnumber, deliverydate) VALUES

('Ajay', '123 Main St', 'Krishna', '456 Elm St', '5', 'Pending', '10', '2024-03-07'),

('Vivek', '456 Elm St', 'Ajay', '789 Oak St', '10', 'Delivered', '20', '2024-03-06'),

('Kowshik', '789 Oak St', 'Niranjan', '101 Pine St', '7', 'In Transit', '30', '2024-03-05'),

('Krishna', '101 Pine St', 'Kowshik', '202 Maple St', '3', 'Pending', '40', '2024-03-07'),

('Vivek', '202 Maple St', 'Krishna', '123 Main St', '8', 'Delivered', '50', '2024-03-06');

-- Inserting sample data into the payment table

INSERT INTO payment (amount, payment\_date, location\_id, courier\_id) VALUES

(50.00, '2024-03-07', 1, 1),

(60.00, '2024-03-06', 2, 2),

(70.00, '2024-03-05', 3, 3),

(80.00, '2024-03-07', 4, 4),

(90.00, '2024-03-06', 5, 5);

-- Inserting sample data into the payment table

INSERT INTO employee (name, email, contact\_num, role, salary, location\_location\_id) VALUES

('Ajay', 'ajay@gmail.com', '1234567890', 'Manager', 50000, 1),

('Krishna', 'krishna@gmail.com', '9876543210', 'Driver', 40000, 2),

('Vivek', 'vivek@gmail.com', '3456789012', 'Warehouse Staff', 35000, 3),

('Niranjan', 'niranjan@gmail.com', '5678901234', 'Courier', 45000, 1),

('Kowshik', 'kowshik@gmail.com', '6789012345', 'Dispatcher', 38000, 2);

INSERT INTO location (location\_name, address) VALUES

-- Inserting sample data into the Location table

('Main Office', '123 Main St, City, Country'),

('Warehouse', '456 Elm St, City, Country'),

('Branch Office', '789 Oak St, City, Country'),

('Distribution Center', '101 Pine St, City, Country'),

('Regional Office', '202 Maple St, City, Country');

-- Inserting sample data into the courier\_service table

INSERT INTO courier\_service (service\_name, cost) VALUES

('Standard Delivery', 10.00),

('Express Delivery', 20.00),

('Same-Day Delivery', 30.00),

('Overnight Delivery', 15.00),

('International Delivery', 50.00);

Task 2: Select,Where Solve the following queries in the Schema that you have created above

-- Query to list all customers

select \* from user;

-- Query to list all orders for a specific customer

SELECT \* FROM courier WHERE sender\_name = 'John Doe';

-- Query to list all couriers

select \* from courier;

-- Query to list all packages for a specific order

select \* from courier where tracking\_num = '1001';

-- Query to list all deliveries for a specific courier

select \* from courier where courier\_id = 1;

-- Query to list all undelivered packages

select \* from courier where status != 'delivered';

-- Query to list all packages that are scheduled for delivery today

select \* from courier where delivery\_date = curdate();

-- Query to list all packages with a specific status

select \* from courier where status = 'pending';

-- Query to calculate the total number of packages for each courier

select courier\_id, count(\*) as total\_packages from courier group by courier\_id;

-- Query to find the average delivery time for each courier

select courier\_id, avg(datediff(delivery\_date, curdate())) as avg\_delivery\_time from courier group by courier\_id;

-- Query to list all packages with a specific weight range

select \* from courier where weight between 5 and 10;

-- Query to retrieve employees whose names contain 'John'

select \* from employee where name like '%john%';

-- Query to retrieve all courier records with payments greater than $50

select c.\* from courier c

join payment p on c.courier\_id = p.courier\_courier\_id

where p.amount > 50;

Task 3: GroupBy, Aggregate Functions, Having, Order By, where

---- Find the total number of couriers handled by each employee.

select e.name, count(c.courier\_id) as total\_couriers\_handled

from employee e

left join courier c on e.employee\_id = c.employee\_id

group by e.name;

----- Calculate the total revenue generated by each location

select l.location\_name, sum(p.amount) as total\_revenue

from location l

left join payment p on l.location\_id = p.location\_id

group by l.location\_name;

----Find the total number of couriers delivered to each location

select l.location\_name, count(c.courier\_id) as total\_couriers\_delivered

from location l

left join courier c on l.location\_id = c.delivery\_location\_id

group by l.location\_name;

------ Find the courier with the highest average delivery time:

select c.courier\_id, avg(c.delivery\_time) as avg\_delivery\_time

from courier c

group by c.courier\_id

order by avg\_delivery\_time desc

limit 1;

---- Find Locations with Total Payments Less Than a Certain Amount

select location\_id, sum(amount) as total\_amount

from payment

group by location\_id

having total\_amount < 5000;

------- Calculate Total Payments per Location

select location\_id, sum(amount) as total\_amount

from payment

group by location\_id;

------ Retrieve couriers who have received payments totaling more than $1000 in a specific location

select courier\_id, sum(amount) as total\_amount

from payment

where location\_id = x

group by courier\_id

having total\_amount > 1000;

---- Retrieve couriers who have received payments totaling more than $1000 after a certain date

select courier\_id, sum(amount) as total\_amount

from payment

where payment\_date > 'yyyy-mm-dd'

group by courier\_id

having total\_amount > 1000;

----Retrieve locations where the total amount received is more than $5000 before a certain date

select location\_id, sum(amount) as total\_amount

from payment

where payment\_date > 'yyyy-mm-dd'

group by location\_id

having total\_amount > 5000;

Task 4: Inner Join,Full Outer Join, Cross Join, Left Outer Join,Right Outer Join

---Retrieve Payments with Courier Information

select p.\*, c.\*

from payment p

inner join courier c on p.courier\_id = c.courier\_id;

---Retrieve Payments with Location Information

select p.\*, l.\*

from payment p

inner join location l on p.location\_id = l.location\_id;

---Retrieve Payments with Courier and Location Information

select p.\*, c.\*, l.\*

from payment p

inner join courier c on p.courier\_id = c.courier\_id

inner join location l on p.location\_id = l.location\_id;

---List all payments with courier details

select p.\*, c.\*

from payment p

left join courier c on p.courier\_id = c.courier\_id;

---- Total payments received for each courier

select c.courier\_id, sum(p.amount) as total\_payments

from courier c

left join payment p on c.courier\_id = p.courier\_id

group by c.courier\_id;

--- List payments made on a specific date

select \*

from payment

where payment\_date = '2024-03-07';

--- Get Courier Information for Each Payment

select p.\*, c.\*

from payment p

left join courier c on p.courier\_id = c.courier\_id;

---- Get Payment Details with Location

select p.\*, l.\*

from payment p

left join location l on p.location\_id = l.location\_id;

--- Calculating Total Payments for Each Courier

select c.courier\_id, sum(p.amount) as total\_payments

from courier c

left join payment p on c.courier\_id = p.courier\_id

group by c.courier\_id;

--- List Payments Within a Date Range

select \*

from payment

where payment\_date between 'start\_date' and 'end\_date';

---- Retrieve a list of all users and their corresponding courier records, including cases where there are no matches on either side

select \*

from user u

left join courier c on u.user\_id = c.user\_id;

---- Retrieve a list of all couriers and their corresponding services, including cases where there are no matches on either side

select \*

from courier c

left join courier\_service cs on c.courier\_service\_id = cs.courier\_service\_id;

--- Retrieve a list of all employees and their corresponding payments, including cases where there are no matches on either side

select \*

from employee e

left join payment p on e.employee\_id = p.employee\_id;

--- List all users and all courier services, showing all possible combinations.

select \*

from user

cross join courier\_service;

---- List all employees and all locations, showing all possible combinations:

select \* from employee

cross join location;

--- Retrieve a list of couriers and their corresponding sender information

select c.\*, s.\*

from courier c

left join (select distinct sendername, senderaddress from courier) s on c.sendername = s.sendername and c.senderaddress = s.senderaddress;

--- Retrieve a list of couriers and their corresponding receiver information

select c.\*, r.\*

from courier c

left join (select receivername, receiveraddress from courier) r on c.receivername = r.receivername and c.receiveraddress = r.receiveraddress;

----Retrieve a list of couriers along with the courier service details (if available):

select c.\*, cs.\*

from courier c

left join courier\_service cs on c.courier\_service\_id = cs.courier\_service\_id;

--- Retrieve a list of employees and the number of couriers assigned to each employee:

select e.employee\_id, e.name, count(c.courier\_id) as num\_couriers\_assigned

from employee e

left join courier c on e.employee\_id = c.employee\_id

group by e.employee\_id;

----- Retrieve a list of locations and the total payment amount received at each location:

select l.location\_id, l.location\_name, sum(p.amount) as total\_payment\_amount

from location l

left join payment p on l.location\_id = p.location\_id

group by l.location\_id;

-----Retrieve all couriers sent by the same sender

select c1.\*

from courier c1

join courier c2 on c1.sendername = c2.sendername

where c1.courier\_id != c2.courier\_id;

----- List all employees who share the same role.

select e1.\*

from employee e1

join employee e2 on e1.role = e2.role

where e1.employee\_id != e2.employee\_id;

---Retrieve all payments made for couriers sent from the same location.

select p1.\*

from payment p1

join payment p2 on p1.location\_id = p2.location\_id

where p1.payment\_id != p2.payment\_id;

-----Retrieve all couriers sent from the same location

select c1.\*

from courier c1

join courier c2 on c1.senderaddress = c2.senderaddress

where c1.courier\_id != c2.courier\_id;

-----List employees and the number of couriers they have delivered:

select e.employee\_id, e.name, count(c.courier\_id) as num\_couriers\_delivered

from employee e

left join courier c on e.employee\_id = c.delivered\_by\_employee\_id

group by e.employee\_id;

------- Find couriers that were paid an amount greater than the cost of their respective courier services

select c.\*

from courier c

join payment p on c