**HELPYAR**

# **Database Management System:**

# **DDL:**

## **Query 1:**

CREATE SCHEMA HELPYAR

## **Query 2:**

CREATE TABLE IF NOT EXISTS User

(

user\_id integer not null unique auto\_increment,

email varchar(50) not null unique,

fname varchar(30) not null,

lname varchar(30),

password varchar(30) not null,

pno varchar(30),

dob date,

age integer,

bio varchar(100),

user\_rank varchar(50),

rating integer,

constraint User\_pk

PRIMARY KEY (user\_id)

);

CREATE TABLE IF NOT EXISTS JobCategory

(

category\_id integer not null unique auto\_increment,

category\_title varchar(50),

category\_type varchar(50),

category\_description varchar(500),

category\_details varchar(1000),

constraint JobCategory\_pk

PRIMARY KEY (category\_id)

);

CREATE TABLE IF NOT EXISTS Job

(

job\_id integer not null unique auto\_increment,

seller\_id integer not null,

title varchar(50),

offered\_price varchar(30),

job\_description varchar(500),

details varchar(1000),

posted\_date date,

job\_assigned varchar(50),

job\_status varchar(50),

job\_type varchar(20),

category\_id integer,

constraint Job\_pk

PRIMARY KEY (job\_id),

constraint Job\_seller\_id\_fk

FOREIGN KEY(seller\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

constraint Job\_category\_id\_fk

FOREIGN KEY(category\_id)

references JobCategory (category\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS worksOnJob

(

job\_id integer not null ,

buyer\_id integer not null ,

CONSTRAINT WorksOn\_pk

PRIMARY KEY(job\_id, buyer\_id),

CONSTRAINT WorksOn\_job\_id\_fk

FOREIGN KEY(job\_id)

references Job(job\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT WorksOn\_buyer\_id\_fk

FOREIGN KEY(buyer\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS Payment

(

job\_id integer not null unique,

amount integer,

payment\_option varchar(50),

payment\_status varchar(20),

constraint Payment\_pk

PRIMARY KEY (job\_id),

CONSTRAINT Payment\_job\_id\_fk

FOREIGN KEY(job\_id)

references Job(job\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS Message

(

message\_id integer not null unique auto\_increment,

message varchar(10000),

message\_time TIMESTAMP,

message\_file varchar(500),

constraint Message\_pk

PRIMARY KEY (message\_id)

);

CREATE TABLE IF NOT EXISTS chatMessage

(

message\_id integer not null unique auto\_increment,

sender\_id integer not null,

reciever\_id integer not null,

constraint Chat\_pk

PRIMARY KEY (message\_id),

constraint Chat\_message\_id\_fk

FOREIGN KEY(message\_id)

references Message(message\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

constraint Chat\_sender\_id\_fk

FOREIGN KEY(sender\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

constraint Chat\_reciever\_id\_fk

FOREIGN KEY(reciever\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS Review

(

review\_id integer not null unique auto\_increment,

reviewer\_id integer not null,

reviewed\_to\_id integer not null,

job\_id integer not null,

rating integer,

review varchar(10000),

constraint Review\_pk

PRIMARY KEY (review\_id)

);

CREATE TABLE IF NOT EXISTS reviews

(

review\_id integer not null unique auto\_increment,

reviewer\_id integer not null,

reviewed\_to\_id integer not null,

job\_id integer not null,

constraint rewiews\_pk

PRIMARY KEY (review\_id),

CONSTRAINT reviews\_review\_id\_fk

FOREIGN KEY(review\_id)

references Review(review\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT reviews\_reviewer\_id\_fk

FOREIGN KEY(reviewer\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT reviews\_reviewed\_to\_id\_fk

FOREIGN KEY(reviewed\_to\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT reviews\_job\_id\_fk

FOREIGN KEY(job\_id)

references Job(job\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS GigCategory

(

category\_id integer not null unique auto\_increment,

categry\_title varchar(50),

category\_type varchar(30),

category\_description varchar(500),

category\_details varchar(1000),

constraint GigCategory\_pk

PRIMARY KEY (category\_id)

);

CREATE TABLE IF NOT EXISTS Gig

(

gig\_id integer not null unique auto\_increment,

gig\_type varchar(30),

title varchar(50),

category\_id integer,

gig\_description varchar(500),

details varchar(1000),

constraint Gig\_pk

PRIMARY KEY (gig\_id),

CONSTRAINT Gig\_category\_id\_fk

FOREIGN KEY(category\_id)

references GigCategory(category\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS hasGig

(

user\_id integer not null,

gig\_id integer not null,

constraint hasGig\_pk

PRIMARY KEY (user\_id, gig\_id),

CONSTRAINT hasGig\_user\_id\_fk

FOREIGN KEY(user\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT hasGig\_gig\_id\_fk

FOREIGN KEY(gig\_id)

references Gig(gig\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS Notification

(

notification\_id integer not null unique auto\_increment,

notification\_text varchar(100),

notification\_time TIMESTAMP,

constraint Notification\_pk

PRIMARY KEY (notification\_id)

);

CREATE TABLE IF NOT EXISTS hasNotification

(

user\_id integer not null,

notification\_id integer not null,

constraint hasNotification\_pk

PRIMARY KEY (user\_id, notification\_id),

CONSTRAINT hasNotification\_user\_id\_fk

FOREIGN KEY(user\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE,

CONSTRAINT hasNotification\_notification\_id\_fk

FOREIGN KEY(notification\_id)

references Notification(notification\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

CREATE TABLE IF NOT EXISTS Search

(

search\_id integer not null unique auto\_increment,

user\_id integer not null,

search\_item varchar(100),

constraint Search\_pk

PRIMARY KEY (search\_id),

constraint Search\_user\_id\_fk

FOREIGN KEY(user\_id)

references User(user\_id)

ON DELETE CASCADE

ON UPDATE CASCADE

);

# **ER DIAGRAM GENERATED DUE TO DDL:**

