CLASSIFIED ADVERTISEMNT SYSTEM (OLX)

DATABASE MANAGEMENT PROJECT

TEAM MEMBERS

Karan Prasad Gupta – 2020439

Shivam Gupta – 2020406

Harsh - 2020434

Scope Of Project

Olx is a classified advertisements website with sections devoted to housing, and second hand products like gadgets, smartphones, cars, furniture, etc. Users can list their item and write a description, upload pictures and set an expected price. Interested users can get in touch with the the seller and negotiate a price and all that through the chat option.

Olx is about:

•giving each other a break, getting the word out about everyday, real-world stuff.
•restoring the human voice to the Internet, in a humane, non-commercial environment.
•keeping things simple, common-sense, down-to-earth, honest, very real.
•providing an alternative to impersonal, big-media sites.
•being inclusive, giving a voice to the disenfranchised, democratizing.
•being a collection of communities with similar spirit, not a single monolithic entity.

INDIVIDUAL CONTRIBUTIONS

Working

1) Login and Register Page:
The user would first need to “Login” or “Register”

2) Home Page:
On the Home Page there will Be 5 options to choose from “Search Listed Items”, “Browse Ads”, “Select Categories”, “Create a New Ad” and “Messages”.

3) Search Listed Items:
If the User clicks on “Search Listed Items” button it will lead the user to a list of all the listed ads people posted on our website for that search. If the user clicks on a specific ad they are interested in, it will lead them to the Ad profile where they could see the description and expected price and other related details and the user can also send message to seller of that specific product from there.

4) Browse Ads:

If the user clicks on “Browse Ads” button it will lead them to most recent ads on our website and then they could just follow the same procedure and negotiate with the seller through messages option.

5) Select Categories:

There are 7 Major Categories in our website. After selecting on a particular category ads will be shown only accordance with your desired field

6) Create a New Ad:

After clicking on “Create a New Ad” button it will lead the user to a window where they can list any products from the available categories. This button will open a window where the user can add title, picture, contact details, description and expected price for the product they wish to sell. The user would also have an option to choose whether or not to display their contact details for this ad. After this it will show under browse ads in our website.

7) Messages:

After Clicking on “Messages” button it will show all the messages in the user’s profile

Stakeholders Identified

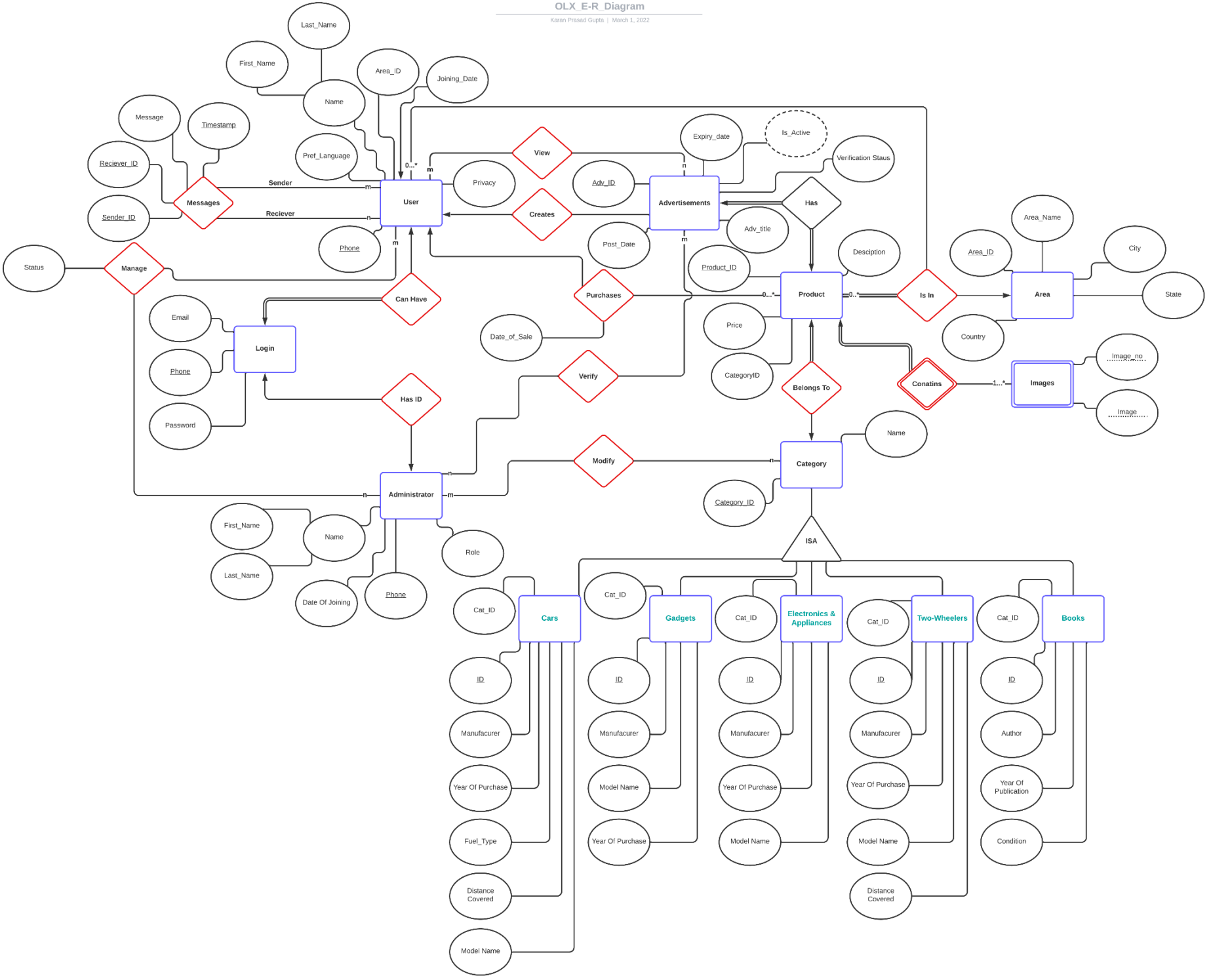
- **Buyers** - They are interested in good quality products at affordable prices. They also want peace of mind with trusted sellers, and trustworthy information in the ads. They also like variety of products..
- **Sellers** - They value their reach to customers, fair charges (our website doesn't charge any charges). They also value more categories, so they can sell different used products.
- **Real Estate Agents** - They also value their reach to the interested customers and ease of posting ads.They should not feel alienated, at best, it increases their usage or the perceived value of listing on our website.
- **Employees**- Income, safe work environment.
- **Government**
- **Owners/Shareholders**

ENTITY RELATIONSHIP DIAGRAM

https://drive.google.com/file/d/1cbOBpbZKp-EBVgjJc-QpPTk6MTAqvqVZ/view?usp=sharing

OLX_E-R_Diagram

Karan Prasad Gupta | March 1, 2022

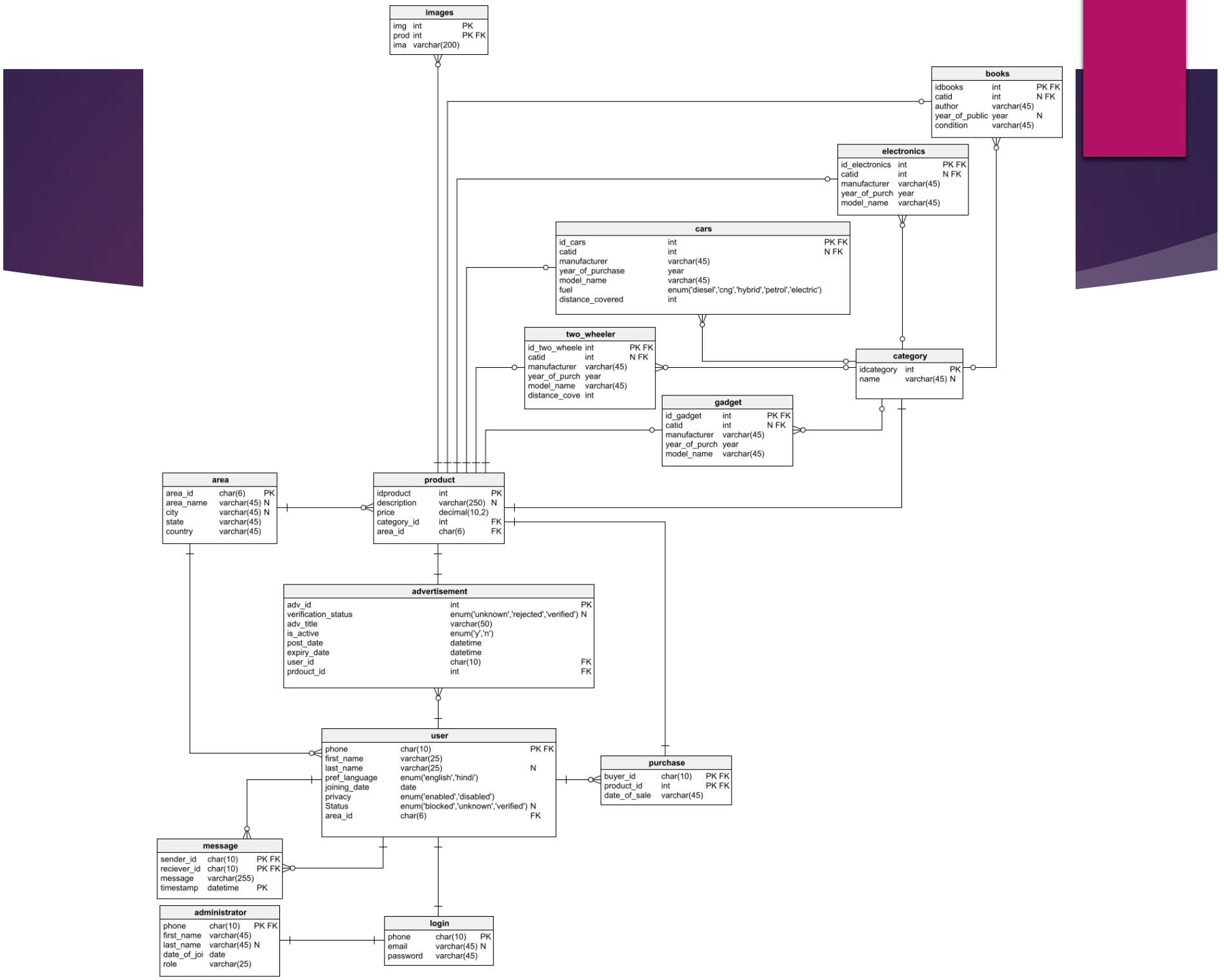


LANGUAGES AND TOOLS REQUIRED

- MySQL Workbench
- Online Data Generator

Relational Schema

https://drive.google.com/file/d/1vDBejGqw8Hm61skCF9Qn9Yda0yvpOxE_/view?usp=sharing


images
img int PK
prod int PK FK
ima varchar(200)
books
idbooks int PK FK
catid int N FK
author varchar(45)
year_of_public year N
condition varchar(45)
electronics
id_electronics int PK FK
catid int N FK
manufacturer varchar(45)
year_of_purch year
model_name varchar(45)
cars
id_cars int PK FK
catid int N FK
manufacturer varchar(45)
year_of_purchase year
model_name varchar(45)
fuel enum('diesel','cng','hybrid','petrol','electric')
distance_covered int
two_wheeler
id_two_wheele int PK FK
catid int N FK
manufacturer varchar(45)
year_of_purch year
model_name varchar(45)
distance_cove int
category
idcategory int PK
name varchar(45) N
gadget
id_gadget int PK FK
catid int N FK
manufacturer varchar(45)
year_of_purch year
model_name varchar(45)
area
area_id char(6) PK
area_name varchar(45) N
city varchar(45) N
state varchar(45)
country varchar(45)
product
idproduct int PK
description varchar(250) N
price decimal(10,2)
category_id int FK
area_id char(6) FK
advertisement
adv_id int PK
verification_status enum('unknown','rejected','verified') N
adv_title varchar(50)
is_active enum('y','n')
post_date datetime
expiry_date datetime
user_id char(10) FK
prdouct_id int FK
user
phone char(10) PK FK
first_name varchar(25)
last_name varchar(25) N
pref_language enum('english','hindi')
joining_date date
privacy enum('enabled','disabled')
Status enum('blocked','unknown','verified') N
area_id char(6) FK
purchase
buyer_id char(10) PK FK
product_id int PK FK
date_of_sale varchar(45)
message
sender_id char(10) PK FK
reciever_id char(10) PK FK
message varchar(255)
timestamp datetime PK
administrator
phone char(10) PK FK
first_name varchar(45)
last_name varchar(45) N
date_of_joi date
role varchar(25)
login
phone char(10) PK
email varchar(45) N
password varchar(45)

Views

```
#v1
#price, ad title, cover image, location, days ago
create view product_view as
select adv.adv_id as "AD ID",adv.adv_title as Title,p.price as
Price,concat(ar.city,', ',ar.state) as
Area,recent_date_calc(adv.post_date) as `Uploaded (days
ago)`,i.image as "Cover Image"
from advertisement as adv, product as p, area as ar, users as u,images
as i
where adv.prdouct_id=p.idproduct and u.phone=adv.user_id and
i.product_id=p.idproduct and i.img_no=1 and u.area_id=ar.area_id
Order by `Uploaded (days ago)`;
```

Views

#v2-count products added evry month in each category

```
CREATE ALGORITHM=UNDEFINED DEFINER=`root`@`localhost` SQL
SECURITY DEFINER VIEW `prodmonthlycount` AS select
month(`a`.`post_date`) AS `Month`,count(0) AS `Ads Posted`,`c`.`name`
AS `Category` from ((`advertisement` `a` join `product` `p`) join
`category` `c`) where ((`a`.`prdouct_id` = `p`.`idproduct`) and
(`c`.`idcategory` = `p`.`category_id`)) group by
`c`.`name`,month(`a`.`post_date`) order by `Month`;
```

Views

#v3- view of all product with all necessary basic details

```
CREATE     ALGORITHM = UNDEFINED     DEFINER = `root`@`localhost`
SQL SECURITY DEFINERVIEW `all_products` AS    SELECT
CONCAT(`u`.`first_name`, ' ', `u`.`last_name`) AS `Product_Owner`,
`ad`.`adv_title` AS `adv_title`,       `ad`.`verification_status` AS
`verification_status`,       `p`.`description` AS `description`,
`cat`.`name` AS `name`,       `p`.`price` AS `price`,       `ad`.`is_active`
AS `is_active`    FROM       (((`product` `p`       JOIN `advertisement` `ad`)
JOIN `users` `u`)       JOIN `category` `cat`)    WHERE       ((`p`.`idproduct`
= `ad`.`prdouct_id`)          AND (`ad`.`user_id` = `u`.`phone`)          AND
(`p`.`category_id` = `cat`.`idcategory`))
```

Grants

```
create role viewer;
GRANT SELECT on olx.all_products TO 'viewer'@'%' WITH GRANT OPTION;
GRANT SELECT on olx.all_products TO 'viewer'@'%' WITH GRANT OPTION;
SHOW GRANTS FOR 'viewer'@'%';

create role dbA;
GRANT ALL PRIVILEGES ON *.* TO 'dbA'@'%' WITH GRANT OPTION;
SHOW GRANTS FOR 'dbA'@'%';
```

Grants

```
create role AdminEmployee;
grant UPDATE, DELETE, SELECT on olx.users TO 'AdminEmployee'@'%' WITH
GRANT OPTION;
grant UPDATE, DELETE, SELECT on olx.advertisement TO
'AdminEmployee'@'%' WITH GRANT OPTION;
grant UPDATE, DELETE, SELECT on olx.product TO 'AdminEmployee'@'%'
WITH GRANT OPTION;
grant SELECT on olx.product_view TO 'AdminEmployee'@'%' WITH GRANT
OPTION;
SHOW GRANTS FOR 'AdminEmployee'@'%';

create role analyst;
grant select on olx.prodmonthlycount to 'analyst'@'%' WITH GRANT
OPTION;
SHOW GRANTS FOR 'analyst'@'%';
FLUSH PRIVILEGES;
```

SQL QUERIES

```
#1
#get alll images
set @product_id=10;
drop table if exists prod_img;
create table prod_img as
SELECT image FROM images where images.product_id=@product_id;
SELECT GROUP_CONCAT(DISTINCT image SEPARATOR '', '') AS img_url
  FROM prod_img;
drop table prod_img;
```

SQL QUERIES

```
#2
#city filter
set @city="delhi";
set @state="";
select *
from product_view
where `Area` like concat('%',@city,'%')  and `Area` like
concat('%',@state,'%')
order by product_view.`Uploaded (days ago)`;
```

SQL QUERIES

```
#3
#select a range of products from a view
set @n=10;
set @rowstart=10;
set @rowend=20;
with temp as (select pr.*,ROW_NUMBER() OVER(order by`Uploaded (days
ago)`) AS `Row No` from product_view as pr)
select * from temp as t
where (t.`Row No`>=@rowstart && t.`Row No`>0) && (t.`Row
No`<=@rowend && t.`Row No`<=(select count(*) from temp));
```

SQL QUERIES

```
#4
#brand filter
SET @brands = '["Mercedes-Benz","Volvo","Honda","Ford"]';
DROP TABLE IF EXISTS brand;
CREATE TABLE brand (WITH RECURSIVE x AS (
    SELECT -1 AS n
    UNION
    SELECT x.n + 1
    FROM x
    WHERE x.n < JSON_LENGTH(@brands) - 1
)
(SELECT (JSON_EXTRACT(@brands, CONCAT('$[', x.n, ']'))) as brand
FROM x
WHERE x.n >= 0));
select c.*
from cars as c, brand as b
where b.brand like concat('%',c.manufacturer,'%') ;
drop table brand;
```

SQL QUERIES

```
#5
#select active and valid ads excluding own ads
set @userid='9132328837';
select pv.*
from product_view as pv inner join advertisement as ad on pv.`AD
ID`=ad.adv_id
where ad.is_active='Y' and DATEDIFF(ad.expiry_date,NOW())>0 and
(ad.prdouct_id not in (select adv.prdouct_id from advertisement as adv
where adv.user_id=@userid));
```

SQL QUERIES

```
#6
#sort by date, price-ltoh, price-htol
set @sorttype="htol";
select *
from product_view as pv
order by
case @sorttype when 'date' then pv.`Uploaded (days ago)` end asc,
case @sorttype when 'ltoh' then pv.`Price` end asc,
case @sorttype when 'htol' then pv.`Price` end desc;
```

SQL QUERIES

```
#7
#offer price- set make offer for a product buyer send a msg to seller
set @userid='9012473368';
set @adid=4;
set @offer_price=93211.00;
set @response = -1;
call olx.`make offer`(@offer_price,@userid, @adid, @response);
select @response;
set @seller= case when @response=1 then (select ad.user_id from
advertisement ad where ad.adv_id=@adid) else NULL end;
insert into message (`sender_id`,
`reciever_id`,
`message`,
`timestamp`)
select@userid,@seller,@offer_price,NOW();
```

SQL QUERIES

```
#8
#search products in similar price range of same category
set @adid=2;
set @prodid= (select ad.prdouct_id from advertisement ad where
ad.adv_id=@adid);
with price_range(min_price,max_price) as
(select 0.7*p.price,1.3*p.price from product as p where
p.idproduct=@prodid),categor(id) as (select p.category_id from product
as p where p.idproduct=@prodid)
select pv.*
from price_range,categor, product_view as pv inner join advertisement
as ad on pv.`AD ID`=ad.adv_id inner join product as pr on
pr.idproduct=ad.prdouct_id
where (pv.Price>=price_range.min_price and
pv.Price<=price_range.max_price) and pr.category_id=categor.id;
```

SQL QUERIES

```
#9
#select blocked users who have posted unverified ads and reject their
ads
set autocommit=0;
update advertisement
set advertisement.verification_status='Rejected'
where advertisement.user_id in (select u.phone
from users as u
where u.Status='Blocked');
SELECT * FROM advertisement inner join users on
advertisement.user_id=users.phone;
rollback;
```

SQL QUERIES

```
#10
#average price of cars of different brands and same fuel type
select cr.manufacturer,cr.fuel,avg(p.price) as `Avg Price`
from cars as cr inner join product as p
where cr.id_cars= p.idproduct
group by cr.manufacturer,cr.fuel
order by cr.fuel,`Avg Price`;
```

Functions & Procedures

```
DELIMITER $$
CREATE PROCEDURE `make offer`(IN offer_price INT,IN buyer CHAR(10),in
adid int, OUT response int)
    READS SQL DATA
BEGIN
     declare product_price int;
     select p.price into product_price
     from product as p where p.idproduct in (select ad.prdouct_id from
advertisement as ad where ad.adv_id=adid);
               IF(offer_price>=(product_price*0.7) and
offer_price<=(product_price*1.3)) then select '1' into response;
     else select product_price into response;
     end if;
end $$
DELIMITER ;
```

Functions & Procedures

```
DELIMITER $$
create function recent_date_calc(datev Datetime )
returns int
DETERMINISTIC
BEGIN
        DECLARE dayago int;
                set dayago=DATEDIFF(NOW(),datev);
        return dayago;
END $$
DELIMITER ;
```

Triggers

```
#t1
DROP TRIGGER IF EXISTS `olx`.`login_BEFORE_INSERT`;

DELIMITER $$
USE `olx`$$
CREATE DEFINER = CURRENT_USER TRIGGER `olx`.`login_BEFORE_INSERT`
BEFORE INSERT ON `login` FOR EACH ROW
BEGIN
        IF NEW.phone not in (Select l.phone from login as l where
NEW.phone=l.phone Or New.email=l.email)
   then insert into `olx`.`login`(phone,email,`password`) values
(New.phone,new.email,new.`password`);
   ElSE
                SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'User With
given Email or Phone No. Already Exists'; END if;
END;$$
DELIMITER ;
```

Triggers

```
#t2
CREATE DEFINER=`root`@`localhost` TRIGGER `cars_BEFORE_INSERT`
BEFORE INSERT ON `cars` FOR EACH ROW BEGIN
            if New.distance_covered<0
   then SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invaild Distance';
   delete from advertisement where
advertisement.prdouct_id=New.id_cars;
   delete from product where product.id_product=New.id_cars;
   else
   INSERT INTO `olx`.`cars`
(`id_cars`, catid`,
`manufacturer`,`year_of_purchase`,`model_name`,`fuel`,`distance_cove
red`)
VALUES
(new.id_cars, '5', new.manufacturer, new.year_of_purchase,
new.model_name, new.fuel, new.distance_covered);
end if;
END
```

Triggers

```
#t3
CREATE DEFINER=`root`@`localhost` TRIGGER `product_BEFORE_INSERT_1`
BEFORE INSERT ON `product` FOR EACH ROW BEGIN
            if New.price<0
    then SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invaild Price';
    else
    INSERT INTO `olx`.`product`
(`idproduct`, `description`, `price`, `category_id`, `area_id`)
VALUES
(new.idproduct,
new.description,
new.price,
new.category_id,
new.area_id);
end if;
END
```

Triggers

```
#t4
DROP TRIGGER IF EXISTS `olx`.`advertisement_BEFORE_INSERT`;
DELIMITER $$
USE `olx`$$
CREATE DEFINER = CURRENT_USER TRIGGER `olx`.`advertisement_BEFORE_INSERT`
BEFORE INSERT ON `advertisement` FOR EACH ROW
BEGIN
            If `olx`.`recent_date_calc`(new.post_date)<>0 or
`olx`.`recent_date_calc`(new.expiry_date)>=0 or
datediff(new.post_date,new.expiry_date)>0
   then
   SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Invaild Dates';
   else
   INSERT INTO `olx`.`advertisement`
(`adv_id`, `verification_status`, `adv_title`, `is_active`, `post_date`,
`expiry_date`,`user_id`, `prdouct_id`)
VALUES(new.adv_id, new.verification_status, new.adv_title, new.is_active,
new.post_date, new.expiry_date, new.user_id, new.prdouct_id);
end if;
END$$
DELIMITER ;
```

Indexing

- Advertisement:

since we frequently access ads posted by same user indexing can be done on (user_id,adv_id)

indexing on title as search based on product title is done

unique

- Area: (city,state) can be used for indexing but we have less data for such tuples so indexing won't be much beneficial
- Books: indexing by author_name can be done for search by author name
- Cars, Gadgets, Electronics, etc..: Indexing by manufacturer name for brand filtering(sparse indexing)
- Images: indexing on (image_no,product_id) but this is already the primary key so indexing not required although sparse indexing on product_id can be done
- Login: emailid is a unique attribute so indexing email for faster searches can be done
- Messages: indexing on (sender_id,receiver_id) to fetch messages faster for 2 ids
- Product: secondary indexing on price, indexing on area_id (can be sparse)

Embedded Queries(PL/SQL)

```
DECLARE
        ID product.idproduct%type = 10;
         Description product.decription%type;
         Price product.price%type;
 BEGIN
        SELECT idproduct, decription, price FROM Product WHERE idproduct = ID;
        GROUP BY ID; HAVING avg(price > 10);
         LIMIT 3;
        dbms_output.put_line('Product' || ID || 'has price' || Price) ;
END;
```

Embedded Queries(PL/SQL)

```
DECLARE
        ID images.image_id%type = 1;
         Product images.product_id%type;
BEGIN
        SELECT product_id, image_id FROM Images WHERE image_id = ID;
        GROUP BY Product;
        LIMIT 5;
        dbms_output.put_line('Image with ' || ID || 'have Product ID' ||
        Product) ;
 END;
```

Embedded Queries(PL/SQL)

```
DECLARE
        ID advertisement.adv_id%type = 1;
        adv_title advertisement.adv_title%type;
BEGIN
        Select u.phone,ad.adv_id,ad.adv_title from users as u advertisement as
        ad where ad.is_active='Y';
        LIMIT 5;
        dbms_output.put_line(advertisement with adv_id ' || ID || 'have
        adv_title' || adv_title) ;
 END;
```

Embedded Queries(PL/SQL)

```
DECLARE
        ID product.idproduct%type = 1;
        price product.price%type;
BEGIN
        select idproduct , price from all_products as ap, where
        ap.price=max_price.maxpr;
        LIMIT 5;
        dbms_output.put_line(product with product_id ' || ID || 'have price' ||
price) ; END;
```

THANK YOU