Part 1

Convert the following EER diagram into a set of tables.

After Step 1

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(make, model)
BikeDeliverer(capacity)
Customers(customerid, name, phone, address, password, email)
Order(ordered, ordertime, orderdate, deliverytime)
Restaurants(restaurantid, name, address, category, website)
FoodItems(menu_name, price, calorieCount)

After Step 2

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(deliveryID, make, model)
BikeDeliverer(deliveryID, capacity)
Customers(customerid, name, phone, address, password, email)
Order(ordered, ordertime, orderdate, deliverytime)
Restaurants(restaurantid, name, address, category, website)
FoodItems(menu_name, price, calorieCount)
After Step 3

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(deliveryID, make, model)
BikeDeliverer(deliveryID, capacity)
Customers(customerid, name, phone, address, password, email)
Order(ordered, ordertime, orderdate, deliverytime)
Restaurants(restaurantid, name, address, category, website)
FoodItems(restaurantid, menu_name, price, calorieCount)

After Step 4

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(deliveryID, make, model)
BikeDeliverer(deliveryID, capacity)
Customers(customerid, name, phone, address, password, email)
Order(ordered, ordertime, orderdate, deliverytime)
Restaurants(restaurantid, name, address, category, website)
FoodItems(restaurantid, menu_name, price, calorieCount)
Includes(ordered, restaurantid, menu_name, quantity)

After Step 5

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(deliveryID, make, model)
BikeDeliverer(deliveryID, capacity)
Customers(customerid, name, phone, address, password, email)
Order(ordered, ordertime, orderdate, deliverytime, deliveryID, customerID)
Restaurants(restaurantid, name, address, category, website)
FoodItems(restaurantid, menu_name, price, calorieCount)
Includes(ordered, restaurantid, menu_name, quantity)

After Step 7

DeliveryPeople(deliveryID, name, phonenumber)
CarDeliverer(deliveryID, make, model)
BikeDeliverer(deliveryID, capacity)
Customers(customerid, name, phone, address, password, email, referID)
Order(ordered, ordertime, orderdate, deliverytime, deliveryID, customerID)
Restaurants(restaurantid, name, address, category, website)
FoodItems(restaurantid, menu_name, price, calorieCount)
Includes(ordered, restaurantid, menu_name, quantity)
Teammate(deliveryID, teammateID)
Use the following database script to answer all questions in Part 2 and 3.

```
CREATE TABLE Reporter (
    reporterID NUMBER,
    reporterName VARCHAR(30),
    reporterEmail VARCHAR(30),
    reporterSpecialty VARCHAR(20))

CREATE TABLE Editor(
    EditorID NUMBER,
    EditorName VARCHAR(30),
    EditorEmail VARCHAR(30),
    EditorDepartment VARCHAR(20))

CREATE TABLE Article(
    articleID NUMBER,
    reporterID NUMBER,
    headline VARCHAR(30),
    articleDate DATE,
    subject VARCHAR(20),
    wordLength NUMBER,
    followupID NUMBER)

CREATE TABLE Revision(
    EditorID NUMBER,
    articleID NUMBER,
    commentText VARCHAR(255),
    commentDate DATE)
```
Part 2

Modify the script on the previous page in the following ways:

- Change the Create Table commands so that primary keys are added to all tables. The primary key for Reporter is reporterID, the primary key for Editor is EditorID, the primary key for Article is articleID, and the primary key for Revision is EditorID and articleID together.
- Change the Create Table commands so that foreign key constraints are added to the tables. Note that every followupID must match an articleID in another row of the article table.
- Change the Create Table command for Article so that wordLength is between 0 and 10000.
- Change the Create Table command so that deleting an Article record will automatically delete all Revision records with the same articleID.

```sql
CREATE TABLE Reporter (  
    reporterID NUMBER,  
    reporterName VARCHAR(30),  
    reporterEmail VARCHAR(30),  
    reporterSpecialty VARCHAR(20),  
    PRIMARY KEY (reporterID))

CREATE TABLE Editor(  
    EditorID NUMBER,  
    EditorName VARCHAR(30),  
    EditorEmail VARCHAR(30),  
    EditorDepartment VARCHAR(20),  
    PRIMARY KEY (EditorID))
```
CREATE TABLE Article(
  articleID NUMBER,
  reporterID NUMBER,
  headline VARCHAR(30),
  articleDate DATE,
  subject VARCHAR(20),
  wordLength NUMBER,
  followupID NUMBER,
  PRIMARY KEY (articleID),
  FOREIGN KEY (reporterID) REFERENCES Reporter,
  FOREIGN KEY (followupID) REFERENCES Article,
  CHECK wordlength >= 0 AND wordlength <= 10000)

CREATE TABLE Revision(
  EditorID NUMBER ,
  articleID NUMBER,
  commentText VARCHAR(255),
  commentDate DATE,
  PRIMARY KEY (EditorID, articleID) ,
  FOREIGN KEY (articleID) REFERENCES Article ON DELETE CASCADE,
  FOREIGN KEY (EditorID) REFERENCES Editor)
Part 3

A. Write a query to output the reporterName of reporters who have written an article with a subject of “College of Saint Rose”.

```sql
SELECT reporterName
FROM Reporter, Article
WHERE Reporter.reporterID = Article.reporterID AND subject = 'College of Saint Rose'
```

B. Write a query to output, for each article, the articleID, the article date, the headline, the followupID, the followup article’s date, and the followup article’s headline.

```sql
SELECT Article.articleID, Article.articleDate, Article.headline, Article.followupID,
Followup.articleDate, Followup.headline
FROM Article, Article AS Followup
WHERE Article.followupID = Followup.articleID
```

C. Write a query to output the reporterID of reporters that have never written an article.

```sql
SELECT reporterID
FROM Reporter
WHERE reporterID NOT IN
(SELECT reporterID FROM Article)
```

D. Write a query to output the name of each reporter and the number of articles they have written.

```sql
SELECT reporterName, COUNT(*) AS Articles
FROM Reporter, Article
WHERE Reporter.reporterID = Article.reporterID
GROUP BY reporterName
```

E. Write a query to output the name of each reporter and the number of articles they have written. Only include reporters that have written at least 10 articles.

```sql
SELECT reporterName, COUNT(*) AS Articles
FROM Reporter, Article
WHERE Reporter.reporterID = Article.reporterID
GROUP BY reporterName
HAVING COUNT(*) >= 10
```

F. Write a query to output the average word length of articles.

```sql
SELECT AVG(wordlength)
FROM Article
```
G. Write a query to output the name and email of all editors and reporters. There should be just two columns in this output.

```sql
SELECT EditorName, EditorEmail
FROM Editor
UNION
SELECT ReporterName, ReporterEmail
FROM Reporter
```

H. Write a query to output the articleID, headline, subject, editorID, commentText, and commentDate for all articles and revisions. If an article does not have a revision, we should still see that article’s other information in the output.

```sql
SELECT Article.articleID, headline, subject, editorID, commentText, commentDate
FROM Article LEFT JOIN Revision ON Article.articleID = Revision.articleID
```

I. Write a query to output the articleID, articleDate and Subject for the article with the smallest wordLength.

```sql
SELECT articleID, articleDate, subject
FROM Article
WHERE wordLength = (SELECT MIN(wordLength) FROM Article)
```