CSE 180, Final Exam, Fall 2023, Shel Finkelstein

Student Name: ______________________________

Student ID: ______________________________

Final Points:

Part I:	**40**
Part II:	**24**
Part III:	**36**
Total:	**100**

The first Section (Part I) of the Fall 2023 CSE 180 Final is Multiple Choice and is double-sided. Answer all multiple choice questions on your Scantron sheet. You do not have to hand in the first Section of the Exam, but you must hand in the Scantron sheet, with your Name and Student ID filled in (including marking bubbles below) on that Scantron sheet. Please be sure to use a #2 pencil (not #3 or ink) to mark your choices on that Section of the Final.

This separate second Section (Parts II and III) of the Final is not multiple choice and is double-sided, so that you have extra space to write your answers on the facing page **after** each question. Please write your Name and Student ID on that second Section of the Exam, which you must hand in. You may use a #2 pencil or ink implement on this Section of the Exam, but make sure that your answers are clear.

At the end of the Final, please be sure to hand in both your Scantron sheet for the first Section of the Exam and also **this second Section of the Exam**. You must also show your **UCSC ID** when you hand them in. Do not hand in your 8.5 x 11 sheet.

Part II: (24 points, 6 points each)

Question 21: Assume that we have previously created:

- a Persons table, whose Primary Key is SSN, and
- a Houses table, whose Primary Key is houseID.

Write a CREATE statement for another table:

Properties(<u>owner, houseID</u>, datePurchased)

which does everything that is described below:

A Properties tuple indicates that a person (owner) purchased a house (houseID) on the specified datePurchased. Attributes owner and SSN are integers. houseID is an integer in both Houses and Properties; datePurchased is a date.

The Primary Key of Properties is (owner, houseID). In the Properties table that you create, owner should be a Foreign Key corresponding to the Primary Key of Persons (SSN), and houseID should be a Foreign Key corresponding to the Primary Key of Houses (which is also called houseID).

When a person in Persons is deleted, tuples in Properties which that person owns should be deleted. When the Primary Key of a person in Persons is updated, then the tuples in Properties which that person owns should have their Foreign Key values updated the same way.

Deletion of a house in Houses is not permitted if there are any tuples in Properties for that houseID. Also, updating of the houseID for a house in Houses is not permitted if there are any tuples in Properties for that houseID**.**

Answer 21:

Answer 21 (continued):

Question 22: We have the following relations:

Sailors(<u>sid</u>, sname, rating, age)
    // sailor id, sailor name, rating, age
Boats(<u>bid</u>, bname, color)
    // boat id, boat name, color of boat
Reserves(<u>sid, bid, day</u>)
    // sailor id, boat id, and date that sailor sid reserved that boat.

Codd's Relational Algebra included only 5 operators, (σ , π , x , U , and -). Write the following query using Codd's Relational Algebra (<u>not</u> SQL):

> *Find the bid and color of the boats which have been reserved by a sailor whose rating is 2, and which have also been reserved by a sailor whose rating is 8.*

If you'd like, you may also use Rename (ρ) and Assignment ($\leftarrow$) in your answer.

To simplify notation, you may write SIGMA for σ and PI for π. You may also use <-- for Assignment and RHO for ρ (Rename). Also, although you can use subscripts, you can also use square brackets, for example writing:

PI[Sailors.sid] (SIGMA[Sailors.age = 21] (Sailors))

instead of writing:

$\pi_{\text{Sailors.sid}}$ ($\sigma_{\text{Sailors.age =21}}$ (Sailors))

Answer 22:

Answer 22 (continued):

Question 23: Answer each of these questions with either the full word **TRUE** or the full word **FALSE**, writing your answers clearly above the blanks using capital letters. No explanation is required, and there will be no part credit.

23a): If myRateFunction is a PL/pgsql Stored Function that takes an integer parameter and returns an integer, you can execute the following at the PostgreSQL psql prompt to see the result of myRateFunction on the value 12.

```
SELECT myRateFunction(12);
```

Answer 23a): ________________

23b): If your PL/pgSQL Stored Function includes the declaration:

```
DECLARE c CURSOR FOR <query>;
```

where <query> is a SQL query, then you must OPEN cursor c before accessing the results of the query by using FETCH commands.

Answer 23b): ________________

23c): In a C program using libpq to access a PostgreSQL database, the following would be a legal statement, assuming that Sells(bar, beer, price) is a table in that database.

```
PGresult *res = PQexec("SELECT price FROM Sells WHERE beer = 'Bud' ");
```

Answer 23c): ________________

23d): In a C program using libpq to access a PostgreSQL database, assume that res is the result set from executing a SQL query on that database. We would write:

```
PQgetvalue(res, 3, 1)
```

to get the value of the first attribute of the third row in that result set,

Answer 23d): ________________

Question 24: This question has two parts; be sure to answer both of them.

Suppose that you have a relation Departments(deptName, building, floor, manager), which has just the following non-trivial Functional Dependencies.

building, floor → deptName
building, floor → manager
deptName → building
manager, building → floor

24a): Is the relation Departments with these Functional Dependencies in **Boyce-Codd Normal Form**? Give a clear detailed proof of your answer.

Answer 24a)

[This is the same relation Departments that was on the previous page, with the same Functional Dependencies.]

Departments(deptName, building, floor, manager)

with just the following non-trivial Functional Dependencies:

building, floor → deptName
building, floor → manager
deptName → building
manager, building → floor

24b): Is the relation Departments with these Functional Dependencies in **Third Normal Form**? Give a clear detailed proof of your answer.

Answer 24b)

Part III: (36 points, 9 points each)

Some familiar tables appear below, with Primary Keys underlined. These tables appear briefly at the top of all 4 questions in Part III of the Final.

SubscriptionKinds(subscriptionMode, subscriptionInterval, rate, stillOffered)

Editions(editionDate, numArticles, numPages)

Subscribers(subscriberPhone, subscriberName, subscriberAddress)

Subscriptions(subscriberPhone, subscriptionStartDate, subscriptionMode, subscriptionInterval, paymentReceived)

Holds(subscriberPhone, subscriptionStartDate, holdStartDate, holdEndDate)

Articles(editionDate, articleNum, articleAuthor, articlePage)

ReadArticles(subscriberPhone, editionDate, articleNum, readInterval)

Assume that no attributes can be NULL, and that there are no UNIQUE constraints. Data types aren't shown to keep thing simple. There aren't any trick questions about data types.

You should assume Foreign Keys as follows:

- Every subscriberPhone in Subscriptions appears as a subscriberPhone in Subscribers.
- Every (subscriptionMode, subscriptionInterval) in Subscriptions appears as a (subscriptionMode, subscriptionInterval) in SubscriptionKinds.
- Every (subscriptionStartDate, subscriptionMode) in Holds appears as a (subscriptionStartDate, subscriptionMode) in Subscriptions.
- Every editionDate in Articles appears as an editionDate in Editions.
- Every subscriberPhone in ReadArticles appears as a subscriberPhone in Subscribers.
- Every (editionDate, articleNum) in ReadArticles appears as an (editionDate, articleNum) in Articles.

Write legal SQL for Questions 25-28. If you want to create and then use views to answer these questions, that's okay, but views are not required unless the question asks for them.

Don't use DISTINCT in your queries unless it's necessary, 1 point will be deducted if you use DISTINCT when you don't have to do so. (Of course, points will also be deducted if DISTINCT is needed and you don't use it.) And some points may be deducted for queries that are over-complicated, even if they are correct.

SubscriptionKinds(subscriptionMode, subscriptionInterval, rate, stillOffered)

Editions(editionDate, numArticles, numPages)

Subscribers(subscriberPhone, subscriberName, subscriberAddress)

Subscriptions(subscriberPhone, subscriptionStartDate, subscriptionMode, subscriptionInterval, paymentReceived)

Holds(subscriberPhone, subscriptionStartDate, holdStartDate, holdEndDate)

Articles(editionDate, articleNum, articleAuthor, articlePage)

ReadArticles(subscriberPhone, editionDate, articleNum, readInterval)

Question 25: paymentReceived is an attribute in Subscriptions which indicates whether payment has been received for that subscription.

Write a SQL statement that defines a view PaidSubscribers. There should be a tuple in PaidSubscribers for each subscriber who has at least one subscription, if payment has been received for every subscription of that subscriber.

The attributes in your result should be called theSubscriberPhone (which should be the phone of the subscriber) and theSubscriberAddress (which should be the address of the subscriber).

No duplicates should appear in your result.

Answer 25:

More space for Answer 25:

SubscriptionKinds(subscriptionMode, subscriptionInterval, rate, stillOffered)

Editions(editionDate, numArticles, numPages)

Subscribers(subscriberPhone, subscriberName, subscriberAddress)

Subscriptions(subscriberPhone, subscriptionStartDate, subscriptionMode, subscriptionInterval, paymentReceived)

Holds(subscriberPhone, subscriptionStartDate, holdStartDate, holdEndDate)

Articles(editionDate, articleNum, articleAuthor, articlePage)

ReadArticles(subscriberPhone, editionDate, articleNum, readInterval)

Question 26: articlePage is an attribute in Articles which indicates the page on which an article appears. readInterval is an attribute in the ReadArticles table which tells us how long a particular subscriber spent reading a particular article.

Write a SQL statement that finds all the articles in Articles:

a) which appear on page 1 in their edition, and
b) where the average time spent by readers of that article is 5 minutes or more.

Your result should include attributes editionDate, articleNum and articleAuthor from the Articles table. Result tuples should appear in reverse alphabetical order based on articleAuthor. If tuples have the same author, tuples with earlier editionDate values should appear before later editionDate values.

No duplicates should appear in your result.

Answer 26:

More space for Answer 26:

SubscriptionKinds(<u>subscriptionMode, subscriptionInterval</u>, rate, stillOffered)

Editions(<u>editionDate</u>, numArticles, numPages)

Subscribers(<u>subscriberPhone</u>, subscriberName, subscriberAddress)

Subscriptions(<u>subscriberPhone, subscriptionStartDate</u>, subscriptionMode, subscriptionInterval, paymentReceived)

Holds(<u>subscriberPhone, subscriptionStartDate, holdStartDate</u>, holdEndDate)

Articles(<u>editionDate, articleNum</u>, articleAuthor, articlePage)

ReadArticles(<u>subscriberPhone, editionDate, articleNum</u>, readInterval)

Question 27: editionDate identifies an edition in the Editions table. numArticles is another attribute in the Editions table.

editionDate is also an attribute in the Articles table. For a particular edition, we can count the number of articles there are in the Articles table for that particular edition. Let's call that the *Computed Article Count* of that edition. It's possible that the *Computed Article Count* for an edition isn't equal to the numArticles value for that edition in the Editions table.

Write a query that outputs just the editions for which the *Computed Article Count* isn't equal to numArticles for that edition. The attributes in your result should be editionDate, numArticles, and computedArticleCount, where computedArticleCount is the *Computed Article Count* for that edition.

No duplicates should appear in your result.

Careful: *There might be some editions for which* the *Computed Article Count is 0, but numArticles is <u>not 0</u>. Since these values aren't equal for such editions, those editions should appear in the result of your query.*

Answer 27:

More space for Answer 27:

SubscriptionKinds(subscriptionMode, subscriptionInterval, rate, stillOffered)

Editions(editionDate, numArticles, numPages)

Subscribers(subscriberPhone, subscriberName, subscriberAddress)

Subscriptions(subscriberPhone, subscriptionStartDate, subscriptionMode, subscriptionInterval, paymentReceived)

Holds(subscriberPhone, subscriptionStartDate, holdStartDate, holdEndDate)

Articles(editionDate, articleNum, articleAuthor, articlePage)

ReadArticles(subscriberPhone, editionDate, articleNum, readInterval)

Question 28: A tuple in Holds is a hold on a particular subscription in Subscriptions. A subscription in Subscriptions corresponds to a subscriber in Subscribers, and to a subscriber kind in SubscriberKinds.

Write a SQL statement that finds all the holds which have all of the following properties:

a) The end date for the hold is December 12, 2023.
b) The end date of the subscription is after January 5, 2024. (You'll have to calculate the subscription's end date.)
c) The name of the subscriber corresponding to that hold has 'usk' (with that capitalization) appearing anywhere in it.
d) The rate of the subscription being held is more than '56.78'.

The attributes which should appear in your result for a hold are the subscriberName, holdStartDate and holdEndDate for that hold. No duplicates should appear in your result.

Answer 28:

More space for Answer 28: