University of Waterloo: Final Examination **Winter Term 2017**

Student Name: _____

Student Signature: _____

Student Identification Number: _____

E&CE 356	Database Systems
Section: 001	Instructor: Paul Ward
Thursday, April 13th 2017	12:30 PM to 3:00 PM
Duration of Exam:	2.5 hours
Number of Exam Pages (including cover sheet):	12 pages / 7 questions
Exam Type:	Closed Book

General Notes

1. No electronic devices including no calculators.
2. Read the entire questions *carefully* before answering.
3. State any assumptions clearly and whenever needed.
4. No questions permitted. When in doubt, make an assumption, write it down.
5. Some SQL reminders are listed on Page 12.
6. *aquila non captat muscas*

Question	**Maximum**	**Score**						**Total**
1	15	(a)		(b)		(c)		
2	15	(a)		(b)		(c)		
3	30	(a)	(b)	(c)	(d)	(e)	(f)	
4	20	(a)			(b)			
5	15	(a)		(b)		(c)		
6	15	(a)			(b)			
7	15	(a)			(b)			
Total	125							

Database for the Final Exam

The following schema and data will be used for several of the questions on this final exam. It is a small database for a car rental company, providing details about cars, customers, and the pickup and dropoff of cars. Neither primary keys nor foreign keys have been specified in this description, and that is deliberate. The table name (relation) is in **bold**.

Customer (CustomerID, Name, City)
Car (Licence, Make, Model, Year)
Pickup (RentalID, CustomerID, Licence, Fee, PickupDate, PickupCity, DropoffDate, Dropoff City)
Dropoff (RentalID, Actual, City, Payment)

Explanation:

- Customer defines a unique customer number, his/her name, and location
- Car defines a unique car, together with information about the vehicle
- Pickup defines any rental contract in which the vehicle has been picked up, when and where, the expected fee, which is based on the expected dropoff date and location
- Dropoff contains the Actual date of dropoff, together with the location of the dropoff, and the Payment. The Payment will be the same as the Fee provided that the vehicle was returned undamaged on the date and at the location specified in the rental contract, as defined in the Pickup relation.

A subset of the data is as shown below:

Customer		
CustomerID int	Name varchar(10)	City varchar(10)
1	Nelson	Waterloo
3	Jimbo	Guelph
4	Moe	Waterloo
5	Lenny	Guelph

Car			
Licence char(8)	Make varchar(10)	Model varchar(10)	Year int
3V3 V3V	Toyota	Corolla	2014
M2M 2M2	Toyota	Corolla	2016
QXW 19Q	Toyota	Camry	2016
ZZZ ZZZZ	Honda	Accord	2014

Pickup							
RentalID int	CustomerID int	Licence char(8)	Fee int	PickupDate date	PickupCity varchar(10)	DropoffDate date	DropoffCity varchar(10)
1	3	ZZZ ZZZZ	150	2017-01-17	Waterloo	2017-01-18	Waterloo
2	5	3V3 V3V	250	2017-01-16	Waterloo	2017-01-19	Waterloo
3	4	M2M 2M2	140	2017-01-21	Kitchener	2017-01-22	Guelph
4	4	ZZZ ZZZZ	180	2017-01-18	Waterloo	2017-01-22	Guelph

Dropoff			
RentalID int	Actual date	City varchar(10)	Payment int
1	2017-01-18	Waterloo	250
2	2017-01-19	Waterloo	250
4	2017-01-23	Guelph	220

1. [15 marks] SQL Comprehension: Considering the database schema and the particular data on page 2, for each of the following queries identify
(i) in a single sentence, what is the query computing?
(ii) what is the output for this particular dataset?

(a) select Model,count(RentalID) as N from Car left outer join Pickup using(Licence) group by Model;

How many times has each car model been rented?

Model	N
Corolla	2
Camry	0
Accord	2

(b) select CustomerID,Name from Customer inner join
(select CustomerID from Pickup where not exists
(select * from Dropoff where Pickup.RentalID=Dropoff.RentalID)) as B
using(CustomerID);

What are the names and IDs of customers who have a vehicle outstanding?

CustomerID	Name
4	Moe

(c) select Licence,Make,Model from Car inner join Pickup using(Licence)
left outer join Dropoff using(RentalID) where Payment is NULL;

Which vehicles (make,model,licence) are outstanding?

Licence	Make	Model
M2M 2M2	Toyota	Corolla

2. [15 marks] Query Creation: Considering the database schema and the particular data on page 2, solve the following queries with *either* Relational Algebra *or* SQL.

(a) What fraction of the time are cars returned later than expected?

```
select Late/Total from
    (select count(RentalID) as Late from Pickup inner join Dropoff using (RentalID)
            where ActualDate-DropoffDate > 0) as LateRentals,
    (select count(RentalID) as Total from Dropoff) as TotalRentals;
```

(b) What is the average cost per day for each Make of vehicle? (For the purpose of this question you may assume subtraction of one date from another will return the number of days different between the two dates)

```
select Make,avg(Payment/(ActualDate-PickupDate)) as CostPerDay from
       Car inner join Pickup using (licence)
           inner join Dropoff using (RentalID) group by Make;
```

(c) How many cars have **definitely** been damaged during their rental period? (Recall that "The Payment will be the same as the Fee provided that the vehicle was returned undamaged on the date and at the location specified in the rental contract.")

```
select count(RentalID) as NumDamaged from
       Dropoff inner join Pickup using (RentalID)
               where Payment > Fee and
                     ActualDate = DropoffDate
                     ActualCity = DropoffCity;
```

3. [30 marks] Keys and Normalization Primary and Foreign Keys were not specified for the car-rental schema on page 2, nor were Functional Dependencies.

(a) What are plausible Primary and Foreign Keys for the four relations?

Customer: Primary Key: CustomerID; no foreign keys
Car: Primary Key: Licence; no foreign keys
Pickup: Primary Key: RentalID; foreign keys: Licence references Car(Licence); CustomerID references Customer(CustomerID)
Dropoff: Primary Key: RentalID; foreign keys: RentalID references Pickup(RentalID)

(b) Knowing that no car manufacturer may call one of their car models with the same name as a competing manufacturer, what Functional Dependencies exist besides those implied by the Primary Keys? Explain.

Model $\rightarrow$ Make

since two different Makes cannot have the same Model.

(c) Is the schema is in either 3NF or BCNF? If so, justify why. If not, explain how to fix it such that it is in one (or both) of those, including specifying any Primary and Foreign Key changes.

No. "Model $\rightarrow$ Make" is a non-trivial FD; Car relation has Licence as key, but Model is not a superset of that (therefore not BCNF) and Make is not a subset of it (therefore not 3NF either).

Fix it by creating relation MakeModel(Make, Model) that contains the mapping from car models to their Make; primary key of this will be Model, and Model is also a foreign key that references Car(Model). The Car relation has the Make attribute removed but it otherwise unchanged. The result is both BCNF and 3NF.

(d) The licence of a vehicle may change (*e.g.,* if it is moved to a different jurisdiction or has a different owner). As such, we want to add the VIN (Vehicle Identification Number) to the database, since it is unique per vehicle and never changes. How should the schema be modified to accommodate this requirement?

Add attribute VIN to the Car relation. Since VIN is unique per Car, it becomes the Primary Key for that relation. Since the Licence is still unique at any given time, the Licence stays in the Car relation but is no longer primary key; it should be identified as unique, though.

Change Pickup and Dropoff so they use VIN rather than Licence; VIN in those relationships would be foreign key referencing Car(VIN).

(e) Phone contact information is missing from the **Customer** relation. Explain how you would add it, using either a single phone contact number or multiple contact numbers.

For a single phone number solution, just add "Phone" attribute to the Customer relation, since CustomerID $\rightarrow$ Phone. If multiple numbers are to be supported, then "CustomerID $\not\rightarrow$ Phone" and so a separate relation CustPhone(CustomerID,Phone) would be needed. In that relation CustomerID would be foreign key referencing Customer(CustomerID). The primary key would either by just Phone or the combination of CustomerID and Phone, depending on whether or not customers were allowed to share phone numbers in the schema.

(f) Are there any other changes that might improve the database schema? Explain.

That's a pretty broad question, but at least one thing does come up within the question thus far: vehicle damage: there is nothing in the Dropoff relation that allows for a vehicle status report.

(Obviously much information is simplified for an exam question; clearly customer address info would be needed; some form of ID, also; likewise, cars would need information about there general state (*e.g.*, if there is any current damage on the vehicle; what colour it is, *etc.*. Pickup and Dropoff would need time-of-day as well as date, and likely more detailed locations than just city names.)

4. [20 marks] Indexing, Hashing, and Query Optimization: Considering the car-rental schema from page 2, we wish to identify where indexes may be of value. Since Primary and Foreign Keys were not specified, and so as to prevent possible errors in the previous question from affecting this question, in your answers identify *all* indexes, even if those indexes are implied by a primary key or foreign key in your previous answers.

(a) Consider the query: select Make,Model from Car natural join Pickup where Fee > 200;
Assuming no indexes, what would the execution plan be and what would be the estimated execution time for that plan if the tables are on disk, in contiguous blocks, the number of rows in Car is r_c, the number of blocks in Car is b_c, the number of rows in Pickup is r_p, the number of blocks in Pickup is b_p, the time to find a random block on disk is T_s and the time to transfer a block from disk is T_t?

Since there are no indexes, the system will do a block nested loop join between the Car and Pickup tables, filtering out any rows where the fee $\not> 200$; the outer table should be whichever has fewer blocks, since those blocks will each require a random lookup and read, while the inner table's blocks can hopefully be read contiguously, after the initial first block is found. Likely the smaller table will be the Car table, since there should be many, many rentals per vehicle, and so it will be the outer table. Therefore, the cost will be:

$(T_s + T_t)b_c + b_c(T_s + b_p T_t)$
$= b_c(2T_s + (1 + b_p)T_t)$

(b) For the query above, identify any indexes over one or more attributes that might potentially improve the query performance. For each index you identify, specify the type of index (B+-tree or Hash or either), whether or not it is a primary or secondary index, if it is a secondary index identify if it is useful if it is an index extension, and justify why the query might benefit from that index.

Since there is a join between Car and Pickup, indexes on the Licence attribute will be useful. Since it is an equi-join, the indexes can be hash or B-tree. For the Car relation, it will be a primary index; for the Pickup relation it will be a secondary index. For the Pickup relation it would not be useful as a secondary index, since it is on the join key, not on the condition.

An index on the fee in the Pickup relation would be useful; this would have to be a B-tree index, since it is a range query which hashes don't support. This would be a secondary index and it would be useful if it was an index extension. Again, it would not be useful if it was an index extension since we have no use of the primary key in the join.

5. [15 marks] Transactions, Concurrency Control, and Recovery: Considering the car-rental schema from page 2, whenever someone is booking a car rental, the availability of the car in the given location must be determined, after which, a new tuple identifying the rental agreement will be added to the **Pickup** relation.

(a) How (precisely) do relational databases ensure that a given car cannot be simultaneously rented to more than one person? Be as specific as possible for this particular scenario; simply stating that they use transactions is not sufficient.

(b) If cars are constantly being rented and returned, any query to compute the total Payments received by the car-rental company could cause problems. What problem(s) could occur? How is this problem solved?

(c) What are the functions of the Write-Ahead Log?

6. [15 marks] Database Design: During the term assignments have been submitted using the Marmoset system. This system is a typical relational-database application with a web front-end. Considering what you know about this from using it, you are required to do an approximate design of the database portion of this application. Specifically, Assignments are defined together with a series of Tests and a due date. Students submit Solutions one or more times, at particular dates/times, which are then tested, and result in a grade.

(a) Create an Entity-Relationship model for the Marmoset system. Your diagram may follow any of the notational conventions described in the lecture notes, in class, or the course textbook, but should be consistent in its usage. Include all relevant entities, attributes and relationships, cardinality constraints, and participation constraints. Indicate primary keys of entities by underlining them.

(b) Translate your ER model into a relational model. You do not need to give specific **create table** commands. However, you should identify the following constraints: primary keys, foreign keys, any attributes which must be unique in a relation, and any attributes which may be NULL. You should also identify any plausible functional dependencies between attributes and ensure that your relational design is normalized or you have justified where you have left it unnormalized.

7. [15 marks] Data Analysis: Consider the data contained within the Marmoset system together with the midterm results. Collectively, they provide a set of data about students including: results on individual questions on assignments and the midterm, number of submission attempts for a question, results of each submission attempt, submission times relative to the due date, *etc.* Students then write a final exam for which they will receive a series of grades for individual questions, which sum together to produce a final grade. Assuming we have the final grade information, as we define a final grade of 80+ as Excellent, 65-79 as Good, 50-64 as Pass, and below 50 as Fail, describe how you would create either a classification tree or a set of associate rules that could be used in the future to predict the final grade for ECE 356 students. Identify how you would validate you approach, together with any limitations.

Some SQL Reminders

DDL:

```
create table
drop table [if exists <table>]
alter table <table> add <attribute> <domain>
alter table <table> drop <attribute>
create view <view> as <query expression>

Domain Types: char(n), varchar(n), int, smallint, numeric(p,d)
              real, double precision, float(n), date, time

Integrity     not null, default <V>, primary key (<a1, ..., an>),
Constraints:  unique(<a1, ..., an>),
              foreign key (<am, ..., an>) references (<am', ..., an'>)

Referential   on update <...>, on delete <...>
Actions:      cascade, set null, set default, no action

Check Constraints: check (<predicate>)
```

DML:

```
insert into <table> values (...)
update <table> set <attribute = ...> where <predicate>
delete from <table> where <predicate>
select <a1,...,an> from <t1,...,tn> where <predicate>

Options on selection attributes: distinct, as

Options on selection tables: natural join, inner join, outer join
                             <join type> on <predicate> using <attributes>

Set Operations: union, interset, except (use 'all' to retain duplicates)

Aggregate Functions: avg, min, max, sum, count

Grouping: group by, having, order by

Options on predicate: =, !=, <, >, <=, >=, in, not in, exists, not exists,
                      is null, is not null, between, like <string pattern>
```

Other Random Reminders

Gini Impurity(S) = $1 - \Sigma_i p_i^2$

Entropy Impurity(S) = $-\Sigma_i p_i log_2(p_i)$