Final Examination
Course 12-324: Database Systems
Exam A Answers

Kinneret College School of Engineering

February 9, 2012 8:30am - 11:30am

- Answer the following questions in English or Hebrew.
- You may bring one double sided page of notes to help you. The page may not contain solutions or answers to previous tests. An appendix with SQL and MS SQL Server formats is attached.
- The number of points for each question is listed next to each one to indicate its weight.
- There are a total of **100** points on the test. You must answer all of the questions.
- Write all of your answers in the test booklet which you received.
- Marks made on the test sheets will not be counted or graded.
- You must return the test questions sheet at the end of the exam.
- You must use proper SQL or MS SQL Server syntax for all answers.

המכללה האקדמית כנרת
בעמק הירדן (ע"ר) בית הספר להנדסה

1 Entity Relational Diagrams (30 points)

Read the following story:

> A contractor performs building projects for customers. Each customer is identified by an id number, a name, and a phone number. Each building project is for one customer and is identified by the project id, the project location, and the total cost. The contractor employs many workers on the projets. Each worker is identified by an id number and a name.
>
> Each project has one or more workers and each worker is assigned to one or more projects. When a worker is assigned to a project, it is recorded in the database along with the start date and end date of the assignment. One worker on each project is assigned to be the foreman. His assignment is recorded in the database with a grade from the customer for his performance as foreman on the project. A worker may be a foreman on many different projects and receive a different grade for each. Some workers aren't foreman on any projects.

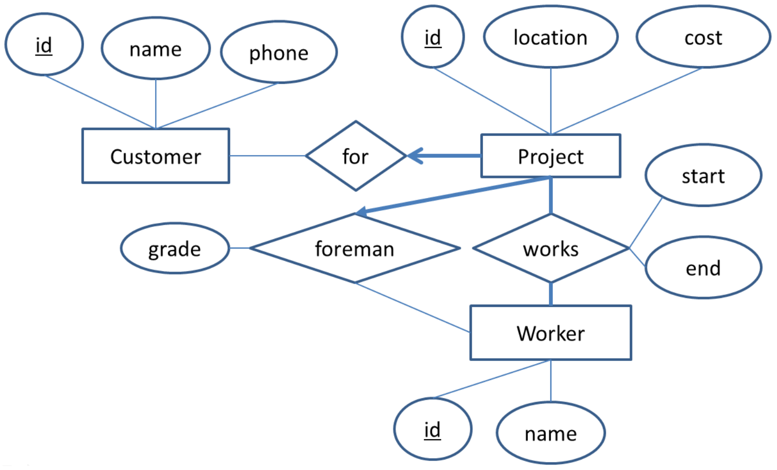
Based on the above story, perform the following actions:

(a) (15 points) Prepare an Entity-Relationship Diagram (ERD) based on the above story. Indicate all attributes, keys, constraints, aggregation, inheritance, or weak entities used (you should not need to use all of them). If you were unable to model some aspect of the story, explain why.

(b) (15 points) Prepare a set of SQL CREATE TABLE definitions based on the ERD you prepared in the previous part. Use correct SQL syntax. Include all primary keys, foreign keys, and constraints. In case you cannot represent a given constraint from the ERD, explain why. Ensure that all of your tables are in the Third Normal Form (3NF).

1.1 Answers

(a) A sample ERD for the above story is:



(b) SQL implementing the above ERD is as follows:

```
CREATE TABLE Customers ( id INT PRIMARY KEY, name VARCHAR(20), phone VARCHAR(12))
CREATE TABLE Projects ( id INT PRIMARY KEY, location VARCHAR(20), cost INT, cid
INT NOT NULL, fid INT NOT NULL, foremanGrade INT, FOREIGN KEY (cid) REFERENCES
Customers, FOREIGN KEY (fid) REFERENCES Workers)
```

```
CREATE TABLE Workers ( id INT PRIMARY KEY, name VARCHAR(30))
CREATE TABLE WorksIn ( pid INT, wid INT, start DATETIME, end DATETIME, FOREIGN
KEY (pid) REFERENCES Projects, FOREIGN KEY (wid) REFERENCES Workers)
```

The "each worker has at least one project" and "each project has at least one worker" constraints were skipped since they are not amenable to SQL tables.

2 Relational Algebra and SQL (45 points)

Consider the following relational schema:

Buses (busId:`INT`, color:`VARCHAR(10)`, numberOfSeats:`INT`, age:`INT`)
Drivers (driverId:`INT`, name:`VARCHAR(30)`, age:`INT`, rating:`INT`, yearsExperience:`INT`)
Routes (routeId:`INT`, start:`VARCHAR(30)`, finish:`VARCHAR(30)`, km:`REAL`)
Drives (routeId:`INT`, busId:`INT`, driverId:`INT`, departTime:`DATETIME`)

2.1 Relational Algebra and SQL (30 points / 10 points each)

Write relational algebra statements and SQL for each of the following queries. You may use string equality (=), LIKE, %, and _ in your Relational Algebra if needed. In all cases, ensure that there are no duplicates in your results.

(a) Show the bus id of all red buses with more than 10 seats.

Algebra $\pi_{(busId)}(\sigma_{(color=`red' \wedge numberOfSeats>10)} Buses)$

SQL

```
SELECT DISTINCT busId FROM Buses B WHERE B.numberOfSeats > 10 AND B.color = 'red'
```

(b) Show names and ages of all drivers who don't drive a route that starts at "Tiberias CBS".

Algebra $\pi_{(name,age)}(Drivers \bowtie (\pi_{(driverId)} Drivers - (\pi_{(driverId)}(Drives \bowtie (\sigma_{(start='TiberiasCBS')} Routes)))))$

SQL

```
SELECT DISTINCT D.name, D.age FROM Drivers D WHERE NOT EXISTS ( SELECT Dr.driverId
FROM Drives Dr1, Routes R1 WHERE Dr1.driverID = D.driverId AND R1.start = 'Tiberias
CBS')
```

(c) Show the names of all drivers who drive all of the routes.

Algebra $\pi_{(name)}(Drivers \bowtie (\pi_{(driverId,routeId)} Drives / (\pi_{(routeId)} Routes)))$

SQL

```
SELECT DISTINCT D.name FROM Drivers D WHERE NOT EXISTS ( SELECT R1.routeId FROM
Routes R1
EXCEPT
SELECT Dr1.routeId FROM Drives Dr1 WHERE Dr1.driverId = D.driverId)
```

2.2 Aggregate SQL Queries (15 points)

Write SQL for the following query:

(a) Show the route id and km of the longest route(s) driven by a driver who doesn't drive any routes which start at "Tiberias CBS"

Answers

```
SELECT R.routeId, R.km FROM Routes R WHERE R.km = ( SELECT MAX(R1.km) FROM Routes
R1, Drives Dr1 WHERE R1.routeId = Dr1.routeId AND Dr1.driverId NOT IN ( SELECT
Dr2.driverId FROM Drives Dr2, Routes R2 WHERE Dr2.routeId = R2.routeId AND R2.start
= 'Tiberias CBS') )
```

3 Triggers (15 points)

Using the above Bus/Drivers/Routes/Drives schema, write a trigger in MS SQL Server syntax which enforces the following condition:

> Create a trigger which will prevent any driver from driving two routes which depart at the same time (two entries in Drives which have the same driver id and depart time).

Make sure to prevent any (and only) INSERT or UPDATE events which would violate the condition, including a case where multiple rows are updated or inserted at once.

3.1 Answers

```
CREATE TRIGGER NoTwoRoutes ON Drives AFTER INSERT, UPDATE AS IF UPDATE(departTime,
driverId) DECLARE @countDb INT; SELECT @countDb = (SELECT COUNT(*) FROM Drives Dr1,
Drives Dr2 WHERE Dr1.driverId = Dr2.driverId AND Dr1.routeId <> Dr2.routeId AND Dr1.departTime
= Dr2.departTime);

IF (@countDb > 0) THEN BEGIN PRINT 'Can't have one driver drive two routes which
depart at the same time' ROLLBACK TRANSACTION END
```

4 Stored Procedures (10 points)

Using the above Bus/Drivers/Routes/Drives schema, write a stored procedure in MS SQL Server syntax which enables the following *parameterized view*:

> Create a stored procedure which shows all of the routes which depart from a given start location (@start). The procedure should show the bus id, the driver's name, the route id, the route's starting location, and the route's finish location.

4.1 Answers

```
CREATE PROCEDURE ShowByStart (@start VARCHAR(30)) AS SELECT Dr.busId, D.name, R.routeId,
R.start, R.finish FROM Drivers D, Routes R, Drives Dr WHERE Dr.driverId = D.driverId
AND Dr.routeId = R.routeId AND R.start = @start
```

5 Appendix: SQL and MS SQL Server Formats

5.1 Basic SQL Queries:

```
SELECT [select-list]
FROM [table list]
WHERE [condition list]
GROUP BY [grouping-list]
HAVING [group-condition]
```

5.2 Basic SQL Data Manipulation:

Insert `INSERT INTO Table1(field1, field2) VALUES (val1, val2)`

Delete `DELETE FROM Table1 WHERE [condition]`

Update `UPDATE Table1 SET [field assignments] WHERE [condition-list]`

5.2.1 Another version for insert:

```
INSERT INTO Table1 (field1, field2)
SELECT f1, f2 FROM Table2
```

5.3 SQL Data Definition:

Table Creation `CREATE TABLE [table name] ([field] [type], ..., PRIMARY KEY ([field list]), FOREIGN KEY ([field list]) REFERENCES [table])`

View Creation `CREATE VIEW [view name] ([field list]) AS [body] [WITH CHECK OPTION]`

5.4 MS SQL Server Stored Procedures and Triggers:

Trigger format:

```
CREATE TRIGGER triggerName ON
   tableName | viewName
   FOR | AFTER | INSTEAD OF
   [INSERT,] [UPDATE,] [DELETE]
   AS [IF UPDATE (columnName)] [batch-code]
```

Stored Procedure format:

```
CREATE PROCEDURE procedureName [(@parameter1
datatype1 [=DEFAULT], [@parameter2 datatype2
[= DEFAULT], ...])]
   AS batch-code
```

5.5 Relational Algebra

The general mapping from relational algebra to SQL:

Algebra	SQL Term
$\pi_{x,y}$	SELECT x, y
$R \times S$	FROM R, S
$\sigma_{x>y}$	WHERE $x > y$
$R \bowtie_c S$	FROM R, S WHERE c
$R \bowtie S$	S NATURAL JOIN R
$\rho(N(p1 \to o_1, p2 \to o_2), E)$	SELECT p_1 AS o_1, p_2 AS o_2 FROM E AS New WHERE …