Click here after you are in full screen to remove this message.

Assessment was due by Tue, 2023-08-22 00:00:00

COMP 421 Midterm 1 Fall 2022 Section 001

Note: There are a total of 100 points on this exams.

Don't panic!

You have 75 minutes to finish the exam.

- You *should* stay in full screen mode.
  - A <ctrl-f> find will give you a warning, which is OK
  - rearranged midterm

You must not leave the window

  - your name when you are ready to submit.
  - Only your **first** submission will be accepted/graded for full credit.
  - If and only if you submit your exam during class you will have a chance to resubmit it for reduced credit. See subsequent submissions on the syllabus for more information.
  - After all students have submitted and the grader has finished, the submit button will be enabled.
  - You have until yyyy-mm-dd hh:mm:ss to submit any subsequent submissions
  - Plan your time judiciously!

I recommend that you have several pieces of scrap paper to doodle notes on during the exam. I *strongly* recommend you read the whole exam and begin with questions you know how to solve quickly. Some questions will be harder or take longer than others; don't spend all your time on one question worth only a few points!

Consider this midterm **closed book**.
You can **NOT** reference other online homeworks, worksheets, etc.
You can use your notes or other things printed out. They should be on paper as you may not switch screens after starting the exam.

You **MAY NOT** Google for anything, You **MAY NOT** leave this website, you **MAY NOT** visit any websites, and you **MAY NOT** copy from a friend. Do not paste information into your midterm unless you know it came from your midterm. You **MAY NOT** receive help from anyone.

If you do not know the origin of material you should not paste it into this exam. All material pasted into this exam must originate from this exam. This implies, but is not limited to, copying from previous assignments, copying from text messages, or copying from **any** website.

You **MUST** use the Google Chrome browser.

The instruction team will **not** answer questions about course content, SQL syntax, etc. We will only deal with issues related to exam implementation.

If your browser hangs, for example because of a bad SQL query, simply kill the page and refresh. It *should* restore all your work even if it doesn't re-evaluate all answers, color-highlight boxes, etc.

You must not leave the window your exam. When you on the screen at the front of the class or given to you by ARS.

You **must not** use your computer or phone in the classroom after you submit your exam. After submitting your exam, simply leave the classroom or ARS.

The browser will change input box color green to indicate correctness. A black or red box indicates an incorrect answer.

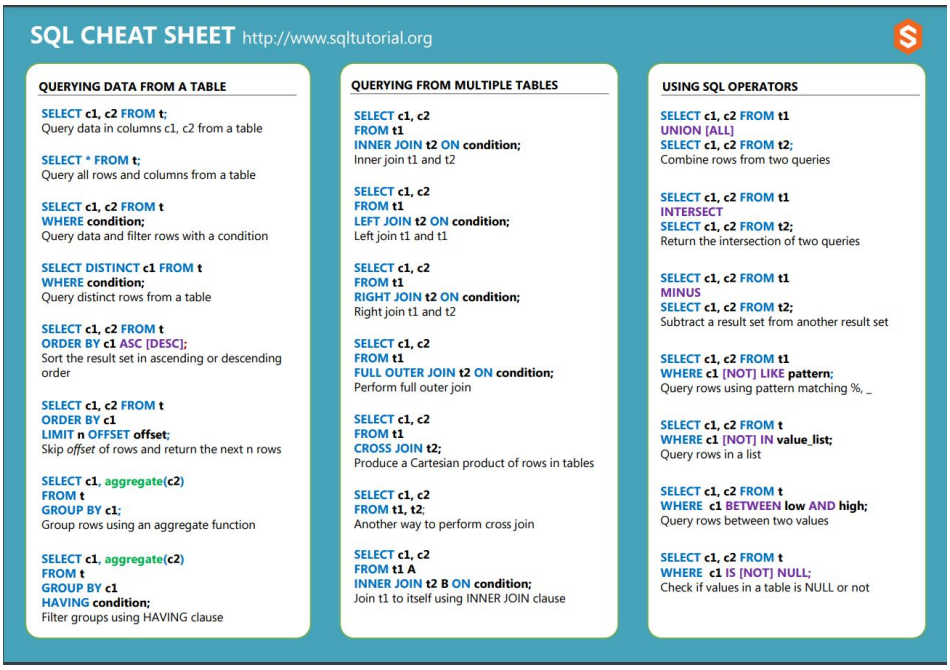
Note that HTML select statements with drop-downs are simple multiple choice questions. No highlighting of correct answers are done for select questions.

Green highlight should just assist you. If you believe your answer is correct and the input box did not turn green, continue on. Per the syllabus, highlighting is simply an aide not a guarantee.

Note: For database queries that are applied to two databases, **two** green lights are required to get any credit for the question.

SQL Tutorial Cheat Sheet

Following are three SQL tutorial cheat sheets available from http://www.sqltutorial.org



SQL CHEAT SHEET http://www.sqltutorial.org

QUERYING DATA FROM A TABLE

SELECT c1, c2 FROM t;
Query data in columns c1, c2 from a table

SELECT * FROM t;
Query all rows and columns from a table

SELECT c1, c2 FROM t
WHERE condition;
Query data and filter rows with a condition

SELECT DISTINCT c1 FROM t
WHERE condition;
Query distinct rows from a table

SELECT c1, c2 FROM t
ORDER BY c1 ASC [DESC];
Sort the result set in ascending or descending order

SELECT c1, c2 FROM t
ORDER BY c1
LIMIT n OFFSET offset;
Skip *offset* of rows and return the next n rows

SELECT c1, aggregate(c2)
FROM t
GROUP BY c1;
Group rows using an aggregate function

SELECT c1, aggregate(c2)
FROM t
GROUP BY c1
HAVING condition;
Filter groups using HAVING clause

QUERYING FROM MULTIPLE TABLES

SELECT c1, c2
FROM t1
INNER JOIN t2 ON condition;
Inner join t1 and t2

SELECT c1, c2
FROM t1
LEFT JOIN t2 ON condition;
Left join t1 and t1

SELECT c1, c2
FROM t1
RIGHT JOIN t2 ON condition;
Right join t1 and t2

SELECT c1, c2
FROM t1
FULL OUTER JOIN t2 ON condition;
Perform full outer join

SELECT c1, c2
FROM t1
CROSS JOIN t2;
Produce a Cartesian product of rows in tables

SELECT c1, c2
FROM t1, t2;
Another way to perform cross join

SELECT c1, c2
FROM t1 A
INNER JOIN t2 B ON condition;
Join t1 to itself using INNER JOIN clause

USING SQL OPERATORS

SELECT c1, c2 FROM t1
UNION [ALL]
SELECT c1, c2 FROM t2;
Combine rows from two queries

SELECT c1, c2 FROM t1
INTERSECT
SELECT c1, c2 FROM t2;
Return the intersection of two queries

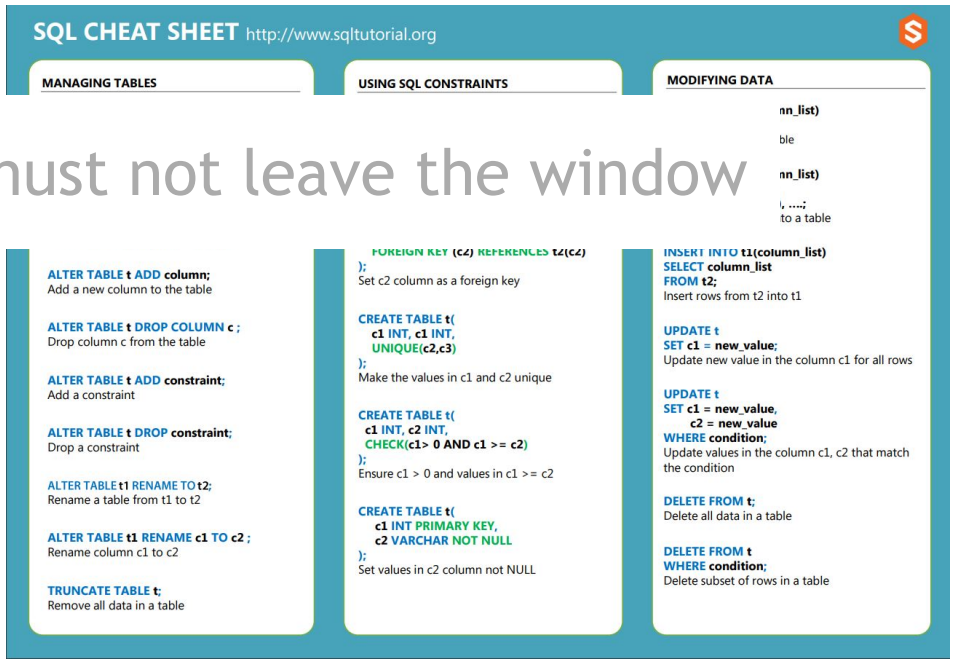
SELECT c1, c2 FROM t1
MINUS
SELECT c1, c2 FROM t2;
Subtract a result set from another result set

SELECT c1, c2 FROM t1
WHERE c1 [NOT] LIKE pattern;
Query rows using pattern matching %, _

SELECT c1, c2 FROM t
WHERE c1 [NOT] IN value_list;
Query rows in a list

SELECT c1, c2 FROM t
WHERE c1 BETWEEN low AND high;
Query rows between two values

SELECT c1, c2 FROM t
WHERE c1 IS [NOT] NULL;
Check if values in a table is NULL or not



SQL CHEAT SHEET http://www.sqltutorial.org

MANAGING TABLES

ALTER TABLE t ADD column;
Add a new column to the table

ALTER TABLE t DROP COLUMN c ;
Drop column c from the table

ALTER TABLE t ADD constraint;
Add a constraint

ALTER TABLE t DROP constraint;
Drop a constraint

ALTER TABLE t1 RENAME TO t2;
Rename a table from t1 to t2

ALTER TABLE t1 RENAME c1 TO c2 ;
Rename column c1 to c2

TRUNCATE TABLE t;
Remove all data in a table

USING SQL CONSTRAINTS

FOREIGN KEY (c2) REFERENCES t2(c2)
);
Set c2 column as a foreign key

CREATE TABLE t(
c1 INT, c1 INT,
UNIQUE(c2,c3)
);
Make the values in c1 and c2 unique

CREATE TABLE t(
c1 INT, c2 INT,
CHECK(c1> 0 AND c1 >= c2)
);
Ensure c1 > 0 and values in c1 >= c2

CREATE TABLE t(
c1 INT PRIMARY KEY,
c2 VARCHAR NOT NULL
);
Set values in c2 column not NULL

MODIFYING DATA

nn_list)

ble

nn_list)

, ...;
to a table

INSERT INTO t1(column_list)
SELECT column_list
FROM t2;
Insert rows from t2 into t1

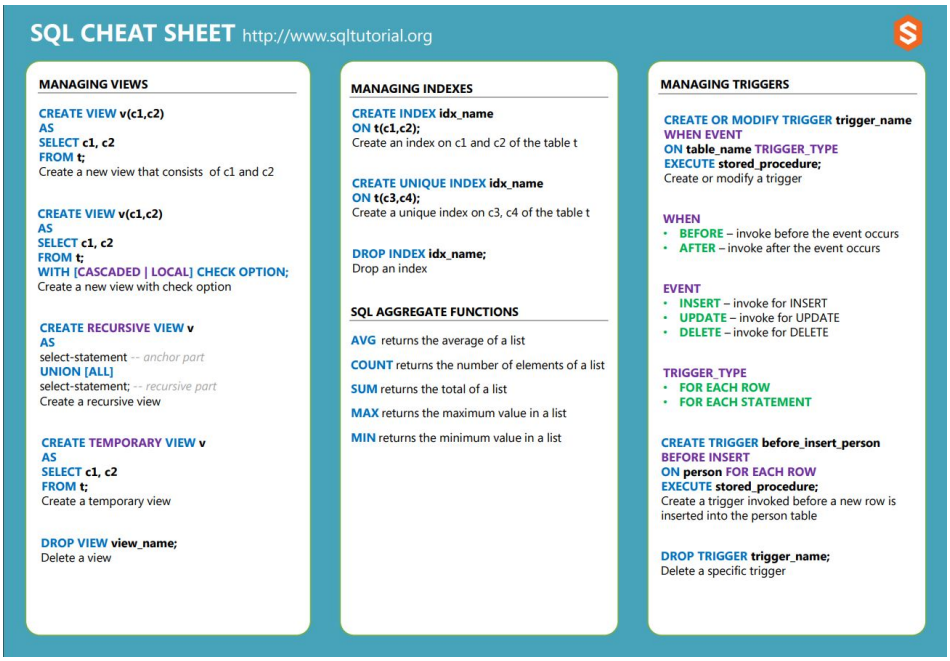
UPDATE t
SET c1 = new_value;
Update new value in the column c1 for all rows

UPDATE t
SET c1 = new_value,
c2 = new_value
WHERE condition;
Update values in the column c1, c2 that match the condition

DELETE FROM t;
Delete all data in a table

DELETE FROM t
WHERE condition;
Delete subset of rows in a table

You must not leave the window



Football Database Schema

Here are the tables you'll find for the database used in the midterm. Your queries will be run against two versions of the database. One of the databases will be much smaller and only contain a subset of the information.

You must not leave the window

```
      name TEXT,
      long_name TEXT)
CREATE TABLE Teams
      (team_id INTEGER,
      school_name TEXT,
      mascot TEXT,
      conf_id INTEGER)
CREATE TABLE Stadiums
      (stadium_id INTEGER,
      name TEXT,
      capacity INTEGER,
      city TEXT,
      state TEXT,
      zip TEXT,
      latitude REAL,
      longitude REAL,
      elevation REAL,
```

```
     team_id INTEGER)
CREATE TABLE Rosters
     (roster_id INTEGER,  -- With nearly 12,000 play
     first_name TEXT,    -- a select of this entire
     last_name TEXT,     -- will take 10-15 seconds!
     height INTEGER,
     weight INTEGER,
     year INTEGER,
     position TEXT,
     city TEXT,
     state TEXT,
     zip TEXT,
     latitude REAL,
     longitude REAL,
     jersey_number INTEGER,
     team_id INTEGER)
CREATE TABLE Games
     (game_id INTEGER,
     home_team_id INTEGER,
     home_team TEXT,
```

You must not leave the window

```
     stadium_id INTEGER,
     completed INTEGER,                -- 0 False, ot
     conference_game INTEGER,          -- 0 False, ot
     attendance INTEGER DEFAULT NULL, -- NULL until
     home_points INTEGER,
     away_points INTEGER,
     week INTEGER)
```

Scratch area

The following scratch space can be used to help develop and test queries against a database described above. The database used by the exam grader will be different.

Execute Minimize Output

Questions For a total of 100 points

SQL Queries 75 points

In this section, you will write SQL queries for the college football schema described at the beginning of the exam. Your queries will be tested immediately against two different databases. If your queries output matches the expected output, the displayed answers will be outlined in green. Your actual score will be determined when your query is tested against a

nat you are on track to

You must not leave the window

op and test queries against these two databases if you want. Nothing about the scratch space contributes to any points on the midterm and anything in the scratch space is ignored.

Scratch space

Execute Minimize Output

List.names.4: Write a SQL command to list each player's full name, otherwise known as their first and last name.

Execute 2.5 points Minimize Output

You must not leave the window

to list the school
'erence' in reverse

alphabetical order by school name.

Execute 5 points Minimize Output

Count.Teams.In.Conferences.4: Write a SQL command to determine how many teams are in the conference with long name 'Big Ten Conference'?

Execute 5 points Minimize Output

Different.Players.4: Write a SQL command that lists the unique home states of players in the conference with long name 'Big Ten Conference'.

You must not leave the window

Execute 10 points Minimize Output

Relational.Algebra.Query.4: Translate this relational algebra expression into a SQL query: $\pi_{name,elevation}\sigma_{zip>80000}Stadiums$

Execute 5 points Minimize Output

Primary.Keys.4: Write a SQL command to create a table named `TeamsII` identical to the current `Teams` table but `TeamsII` should have a primary key of the first field.
Note$_1$: The tests of your code will populate the new tables, list the table, then test the integration constraints. The test of the constraints may issue an UNIQUE error. A correct answer has all green output.
Note$_2$: You may want to prefix your SQL command with a DROP command so that if you re-run your command you avoid getting an error.

You must not leave the window

Execute 10 points Minimize Output

Integrity.Constraint.4: Write a SQL command to create a table named `GamesII` identical to the current `Games` table but `GamesII` should have a primary key of the first field and two foreign keys to `TeamsII`.
Note$_1$: The tests of your code will populate the new tables and test the integration constraints.

Note$_2$: To get this question correct, you must get question **Primary.Keys4** to execute correctly!

Note$_3$: You may want to prefix your SQL command with a DROP command so that if your re-run your command you get an error.

Execute 10 points Minimize Output

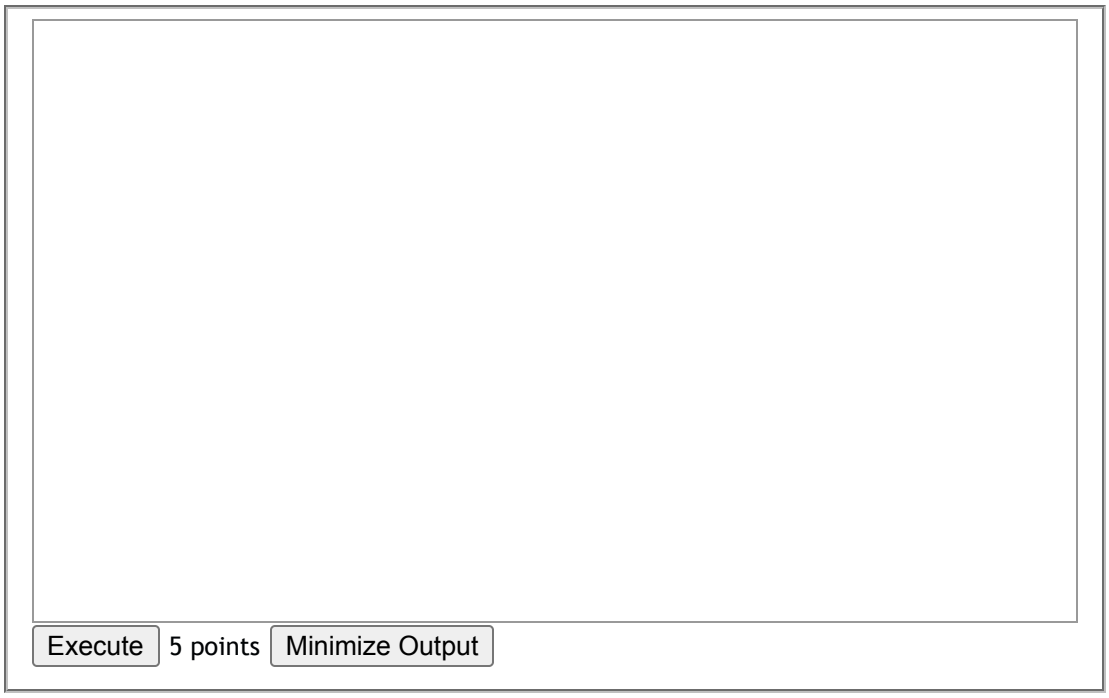
Schools.With.Roster.Size.4: Write a SQL command to list the school names with between 100 and

You must not leave the window

Execute 10 points Minimize Output

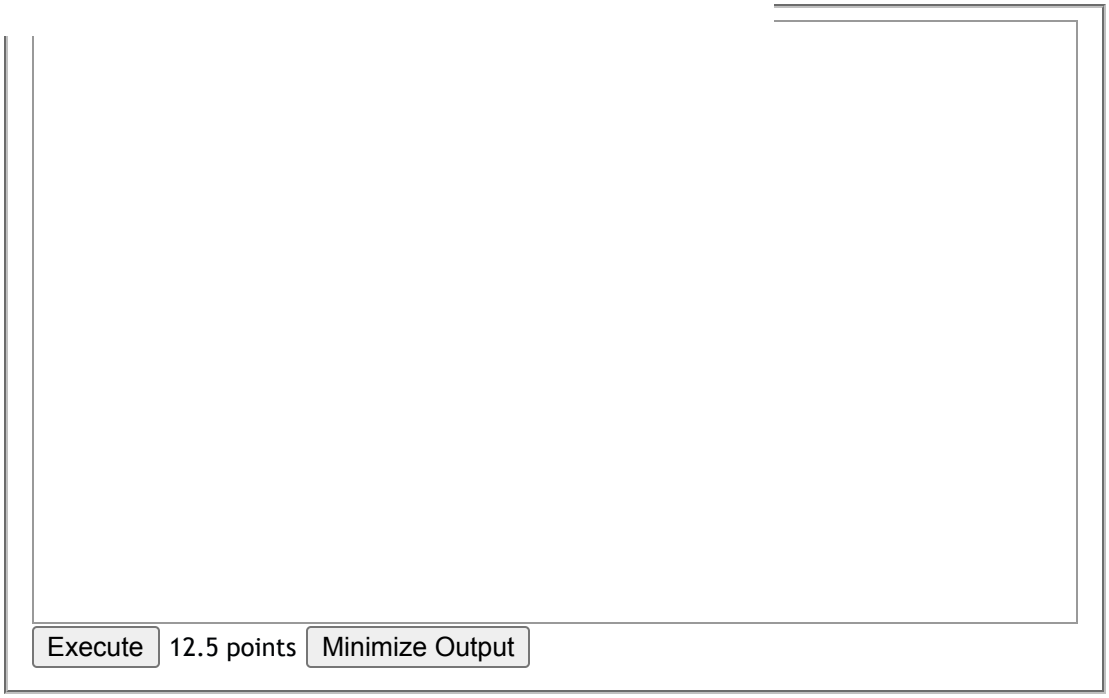
List.Stadiums.4: Write a SQL command to list the stadium names and their latitude for stadiums in the conference with the long name 'Atlantic Coast

Conference' in descending order by stadium name.



List.Games.4: Write a SQL command to list the home teams from the games in the conference with long name 'Atlantic Coast Conference' for the games where the game was held at the farthest north (highest latitude).
NOTE: The question states games and teams, plural, because there could
nd therefore there

You must not leave the window



Chapter Reading Review 25 points

Relational_Algebra.4: What does the SQL relation algebra operator $\bowtie$ do?

5 points

Reserve.Boats.4: Which of the following SQL relational algebra expressions finds the distinct sailor names that reserved a red and green boat.

A: $\rho(TempBoat1, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$
$\rho(TempBoat2, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$
$\pi_{sname}((TempBoat1 \cup Tempboat2) \bowtie Sailors)$

B: $\rho(TempBoats2, (\sigma_{color='red'} Boats) \cup (\sigma_{color='green'} Boats))$
$\pi_{sname}(Tempboats2 \bowtie Reserves \bowtie Sailors)$

C: $\rho(TempBoats2, (\sigma_{color='red'} Boats) \cap (\sigma_{color='green'} Boats))$
$\pi_{sname}(Tempboats2 \bowtie Reserves \bowtie Sailors)$

D: $\rho(TempBoat1, \pi_{sid}((\sigma_{color='red'} Boats) \bowtie Reserves))$
$\rho(TempBoat2, \pi_{sid}((\sigma_{color='green'} Boats) \bowtie Reserves))$
$\pi_{sname}((TempBoat1 \cap Tempboat2) \bowtie Sailors)$

E: None of them.

5 points

SQL_construct.4: What SQL construct ensures certain minimal subset of a relation's fields are unique?

You must not leave the window

5 points

Foreign.Key.4: Which of the following item(s) is a requirement of a foreign key?

5 points

Time elapsed: 0:08