**CSC 430 – Database Management Systems**

Exam 2 - PRACTICE

Name:

Instructions:

* Put your name in the appropriate place at the top of this page;
* Do not use red ink;
* **Closed books and notes;**
* **No electronic devices are allowed; This includes calculators.**
* You will only receive points for a question if you attempt to answer it;
* For full credit, list all formulas that provide the basis for calculations and show all work;
* If you aren't clear about a question, state your assumptions first followed by your answer;
* When finished with the exam read and sign the pledge at the bottom of this page.

Good luck!

# “On my honor as a Louisiana Tech student, I have neither given nor received unauthorized assistance on this academic work.”

Student signature

**Section A: Multiple-choice questions.**

Please, circle a single correct option.

1. Select correct statement(s):
   1. Data Definition Language (DDL) includes commands such as CREATE and DROP.
   2. Data Definition Language (DDL) includes commands such as SELECT and INSERT.
   3. Data Manipulation Language (DML) includes commands such as ALTER and TRUNCATE.
   4. Data Manipulation Language (DML) includes commands such as UPDATE and DELETE.
   5. Only a, b.
   6. Only b, c.
   7. Only c, d.
   8. **Only a, d.**
2. Select correct statement(s) about functional dependencies:
   1. They are not needed for the relation normalization process.
   2. They are determined by the interpretation of the mini-world.
   3. Given the state of a database, we can see if a FD is violated.
   4. All of the above.
   5. Only a, b.
   6. **Only b, c.**
3. Select correct statement(s) about the goal(s) of the normalization process:
   1. A goal is to minimize data anomalies from occuring.
   2. A goal is to make database overly complicated.
   3. A goal is to preserve functional dependencies.
   4. A goal is to maximize the number of prime attributes.
   5. All of the above.
   6. **Only a, c.**
   7. Only b, c.
   8. Only a, b.
4. Select correct statement(s):
   1. Primary indexes are used when data file is ordered by primary key attributes.
   2. Clustering index is used when data file is ordered by a non-key attribute.
   3. Secondary index is defined over non-ordering attributes of a record.
   4. **All of the above.**
   5. None of the above.
   6. **Only a, b, c.** *(this is correct as well)*
   7. Only a, b, d.

**Section B: Open-ended questions.**

To get full points provide complete answer, be specific and concise.

1. Show the full syntax of a trigger.

**CREATE TRIGGER TRIGGER-NAME**

**BEFORE/AFTER UPDATE/DELETE/INSERT**

**ON TABLE-NAME**

**FOR EACH ROW**

**BEGIN**

**SQL STATEMENTS**

**END**

1. Show the full syntax of a function.

**CREATE FUNCTION FUNC-NAME (PARAM-NAME PARAM-TYPE, ...)**

**RETURNS RETURN-TYPE**

**CHARACTERISTIC**

**BEGIN**

**SQL STATEMENTS**

**END**

where CHARACTERISTIC is:

either: **DETERMINISTIC** / **NOT** **DETERMINISTIC**

and either: **READS** **SQL** **DATA** / **MODIFIES** **SQL** **DATA** / **CONTAINS** **SQL** / **NO** **SQL**

1. What are the differences between a trigger, view, stored procedure and a function?

**refer to chart at end of lesson 7.4:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Triggers** | **Views** | **Stored Procedures** | **Functions** |
| Invoked automatically when the trigger event happens | Invoked when you use them in FROM clause | Invoked when you CALL them | Invoked when you use them in SELECT or WHERE clause |
| Can execute any SQL commands | Can only SELECT  (acts as a virtual table) | Can execute any SQL commands | Can SELECT and calculate |
| Can stop an operation from occurring |  | Can take parameters | Can take parameters |
| Can alter what an operation does |  | Can produce many results  (as OUT parameter(s)) | Can only produce a single result (as a returned value) |

1. State the condition(s) for each normal form to be satisfied.

First normal form:

* **All attributes must be have atomic values**
* **All tables must have a primary key**
* **Domains must be consistent within a column/attribute**

Second normal form:

* **Any functional dependencies where a primary attribute is on the left hand side, must include the entire key**
  + **i.e. no partial dependencies**

Third normal form:

* **No functional dependencies can exist between two non-prime attributes**
  + **i.e. no transitive dependencies**

**Section C: Practical questions.**

To get full points show all work, provide all formulas and calculations.

1. Write a query to retrieve Ssn of all employees who worked on project number 2 for at least 15 hours. For full points - use aliasing when joining tables and rename attribute of the resulting relation to “Employee\_SSN”. Database state is provided for you on a separate page.

**SELECT e.Ssn AS Employee\_SSN**

**FROM EMPLOYEE AS e**

**JOIN WORKS\_ON AS w ON e.Ssn = w.Essn**

**WHERE w.Pno = 2 AND w.Hours >= 15;**

1. Write a function to get the sum of all hours across all projects and the count of all projects worked on by a particular department number (given as parameter). You must give correct characteristics to the function for full credit. Database state is provided for you on a separate page.

**DELIMITER $$**

**CREATE FUNCTION GetDepartmentProjectHoursAndCount(department\_number INT)**

**RETURNS VARCHAR(100)**

**DETERMINISTIC**

**READS SQL DATA**

**BEGIN**

**DECLARE total\_hours DECIMAL(10, 2);**

**DECLARE project\_count INT;**

**DECLARE result VARCHAR(100);**

**SELECT SUM(w.Hours) INTO total\_hours**

**FROM WORKS\_ON w**

**JOIN PROJECT p ON w.Pno = p.Pnumber**

**WHERE p.Dnum = department\_number;**

**SELECT COUNT(DISTINCT p.Pnumber) INTO project\_count**

**FROM PROJECT p**

**WHERE p.Dnum = department\_number;**

**SET result = CONCAT('Total Hours: ', total\_hours,**

**', Project Count: ', project\_count);**

**RETURN result;**

**END $$**

**DELIMITER ;**

1. Write a trigger that would remove all projects (from the project table) with the corresponding department number when a row is deleted from the department table. Database state is provided for you on a separate page.

**DELIMITER $$**

**CREATE TRIGGER Delete\_Department\_Projects**

**AFTER DELETE ON DEPARTMENT**

**FOR EACH ROW**

**BEGIN**

**DELETE FROM PROJECT**

**WHERE Dnum = OLD.Dnumber;**

**END $$**

**DELIMITER ;**

1. Define which of the provided functional dependencies may hold for the following relation. If the dependency cannot hold – justify your answer, by specifying at least one tuple that causes the violation.

|  |  |  |  |
| --- | --- | --- | --- |
| **OEM** | **Model** | **Distributor** | **Price** |
| Dell | Optiplex | Office Depot | $1,200 |
| Dell | Optiplex | Dell.com | $1,200 |
| HP | Z Book | hp.com | $2,000 |
| HP | Y Book | Bob in a parking lot | $350 |
| Lenovo | Thinkpad | lenovo.com | $10,000 |
| Lenovo | Ideapad | lenovo.com | $1,500 |

1. OEM -> Model

**Doesn’t hold. Each OEM doesn’t correspond to only one model. (HP -> Z Book and HP -> Y Book )**

1. OEM -> Price

**Doesn’t hold. Each OEM doesn’t correspond to only one price (e.g., HP -> $2000 and HP -> $350)**

1. {OEM, Model} -> Price

**Holds because, for each combination of OEM and Model, the Price is unique.**

1. Price -> Model

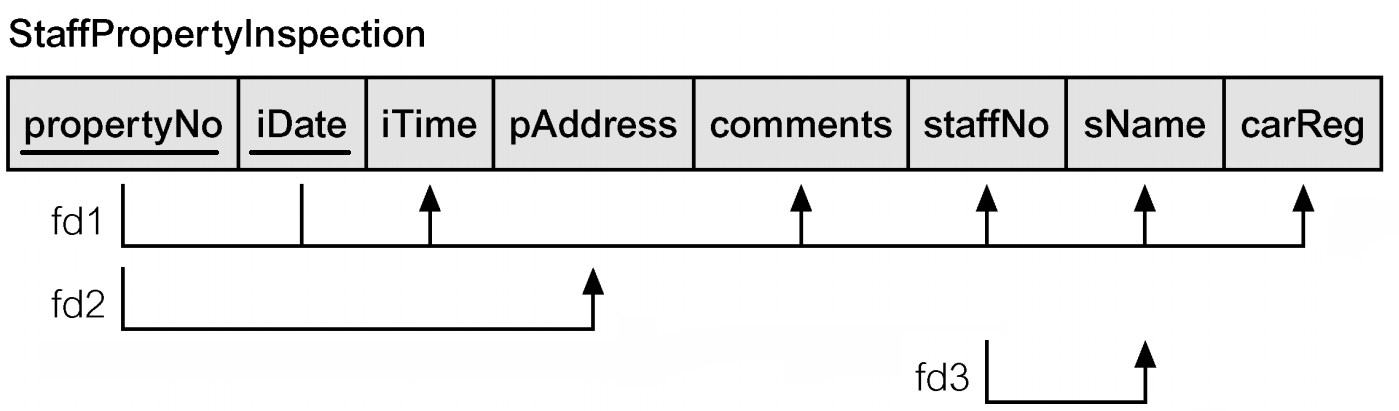
**Holds because each Price corresponds to only one Model.**

1. Price -> Distributor

**Doesn’t hold. Same Price corresponds to multiple Distributors**

**(e.g., $1200 -> Office Depot, $1200 -> Dell.com)**

1. What is the highest normal form of the following relation? To support your answer, specify which functional dependencies violate which of the normal forms.

****

**1NF is the highest form. Although one could argue that "comments" may indicate a non-atomic attribute in which case it would violate 1NF. Your explanation on which FD violates which forms would indicate your understanding of the schema here.**

**FD2 = violates 2NF since we have a partial key dependency.**

**FD3 = violates 3NF since we have two non-prime attributes forming a dependency (which indicates a transitive dependency)**

1. Consider a disk with a block size B = 2048 bytes. Suppose file has r = 8,000 DEPARTMENT records of fixed length. Each record has following attributes: Dname (25 bytes), Dnumber (4 bytes), Mgr\_ssn (9 bytes), Mgr\_start\_date (10 bytes). In addition, suppose the file is sorted on Dnumber primary key attribute. Assuming an unspanned organization, calculate the record size **R**, the size of file ordering key **V**, blocking factor **bfr**, and the number of file blocks **b.** Show all formulas (with variable names), all calculations (with values plugged in), and give final numeric answers for each (i.e. don't leave any final answer as a math equation).

**B** = 2,048 bytes

**r** = 8,000 records

**R** = sum of the sizes of all attributes in record (Dname + Dnumber + Mgr\_ssn + Mgr\_start\_date)

**R** = 25 + 4 + 9 + 10 = 48 bytes

**V** = size of file ordering key (primary key attribute Dnumber**)**

**V** = 4 bytes

**bfr** = = = 42 records per block

**b** = = = 191 blocks

