THE GEORGE WASHINGTON UNIVERSITY

WASHINGTON, DC

6b. HW/Exam Review

CSCI 2541W Database Systems & Team Projects

Parmer (based on Wood)

Today…

Exam Logistics

SQL HW Review

Normalization HW Review

Exam Logistics

Wednesday starting at 12:45PM

Exam will be on computer

- multiple choice
- SQL

Class ends at 3:25PM

- You can use both periods if you need

If you have a disability that affects your ability to complete the exam, contact me ASAP!

You…

may:

- Use 1 page (double sided) notes – cannot share notes
- Use normalization reference – print them out!

may not:

- Use a computer/phone/device to access any material not explicitly allowed by the exam – only the form, or vscode opening only the midterm's code.
- Discuss questions or get help from anyone else
- Do anything else which violates the course or GW's academic integrity policies

Violating these policies will have severe consequences, including **failing** the course

Suggestions

Make your own notes

- Explain the core concepts to yourself by rewriting in your own words
- Writing out your own version of the key rules (2NF vs 3NF, lossless decomposition rules, etc) will help you fully understand them!
- Try to solve the homework problems without looking at solutions

Be an efficient test taker

- Assume you might not finish, an triage
- Focus first on the sections you are most confident with
- Don’t waste too much time on any one question

Schema for Bank database:

Customer (CustID, Name, street, city, zip)

- Customer ID, Name, and Address info: street, city, zip

Deposit (CustID, Acct-num, balance, Branch-name)

- Customer ID, Account number, Balance in account, name of branch where account is held;

CustID is foreign key referencing Customer.

- Branch-name is foreign key referencing Branch relation

Loan (CustID, Loan-num, Amount, Branch-name)

- Customer ID, loan number, amount of loan; CustID is foreign key referencing Customer relation;
- Branch-name is foreign key referencing Branch relation.

Branch (Branch-name, assets, Branch-city)

- Name of the branch (unique name), assets in dollars, and the city where the branch is located.

Next: SQL Queries

Schema for Company DB

Employee

- Connects to Department by Dno

Department

- Connects to Employee with Mgr_ssn

Dept_locations

- Connects to department

Project

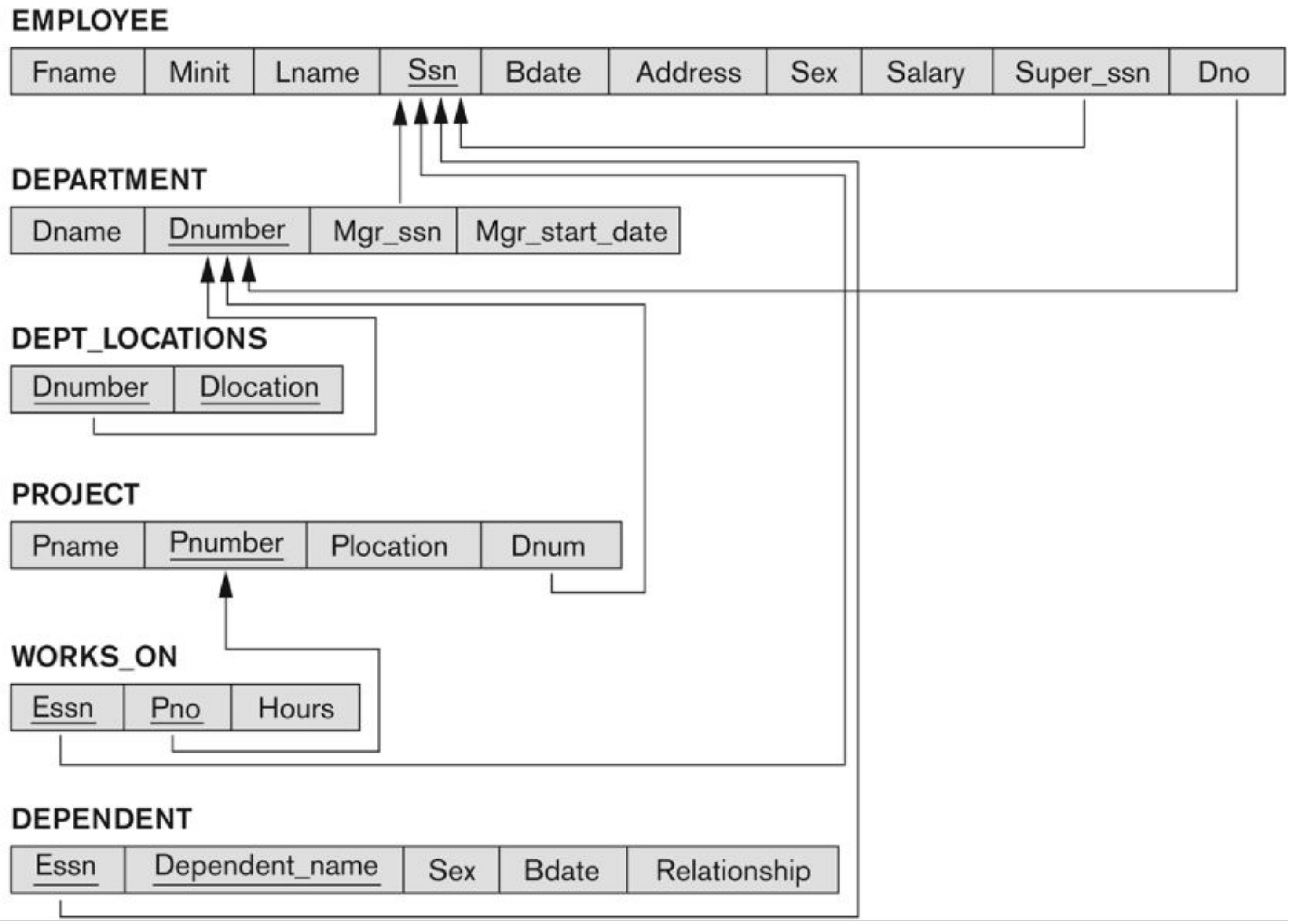
- Connects to Department

Works_On

- Connects from Employee to Project

Dependent

- Connects to Employee



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

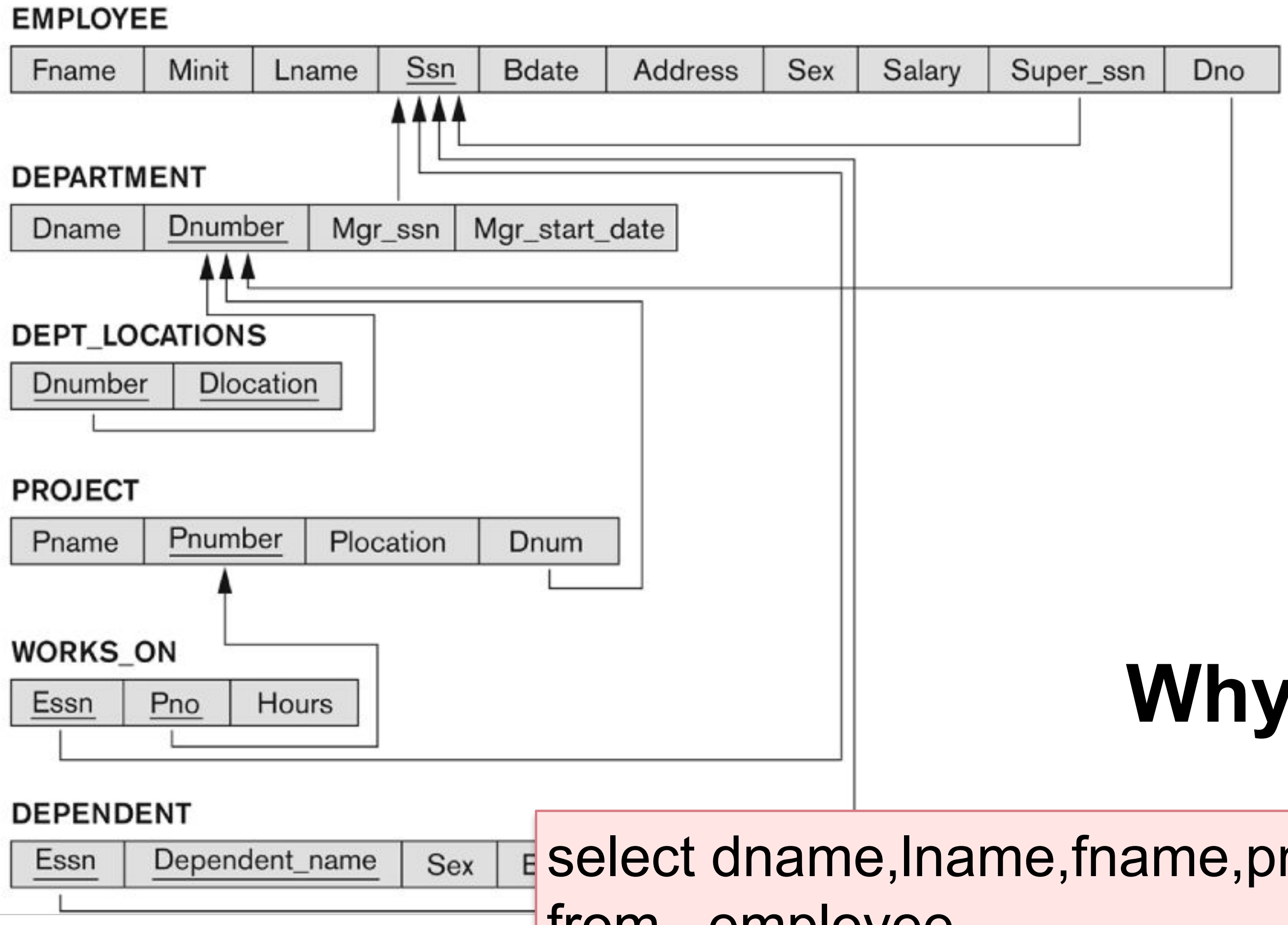
Essn	Pno	Hours

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

SQL HW

7. Retrieve the list of employees, the projects they are working on, and their salary.



Why is this wrong?

```
select dname,lname,fname,pname,salary
from   employee
JOIN department on department.dnumber=employee.dno
JOIN project on project.Dnum = Employee.dno;
```

SQL HW

7. Retrieve the list of employees, the projects they are working on, and their salary.

Just because a project is in a department, doesn’t mean that employee works on it! Need to join using the works_on table.

```sql
select dname,lname,fname,pname,salary
from   department
JOIN employee on department.dnumber=employee.dno
JOIN works_on on works_on.essn = employee.ssn
JOIN project on project.pnumber = works_on.pno
```

Complex Queries

Sometimes you need a subquery within a query

```
SELECT name FROM
instructors
WHERE rating = (
  SELECT rating
  FROM instructors
  WHERE name = 'Wood'
);
```

```
SELECT name
FROM city
WHERE country_id IN (
  SELECT country_id
  FROM country
  WHERE population > 20000000
);
```

Or you need to combine results from queries

- UNION, INTERSECT, EXCEPT

```
SELECT DISTINCT name FROM instructors
EXCEPT
SELECT DISTINCT name from students;
```

Complex Queries

Sometimes you need a subquery within a query

```
SELECT name FROM
instructors
WHERE rating = (
  SELECT rating
  FROM instructors
  WHERE name = 'Wood'
);
```

Only a single result

```
SELECT name
FROM city
WHERE country_id IN (
  SELECT country_id
  FROM country
  WHERE population > [illegible]0000000
);
```

“Set” of results
NOT IN works too!

Or you need to combine results from queries

– UNION, INTERSECT, EXCEPT

```
SELECT DISTINCT name FROM instructors
EXCEPT
SELECT DISTINCT name from students;
```

Practice!

Review HW3!

Engage!

- Write a DB query problem and post on Discord in #participation-points!

Any other questions on SQL?

Next: Normalization

Normal Forms - more definitions

2NF: A schema is in 2NF if

- No nonprime attribute is partially dependent on the candidate key (i.e., depends on only part of a candidate key)
- *No dependencies from a subset of the primary key*

3NF: A schema is in 3NF if (it is 2NF and)

- no nonprime attribute is transitively dependent on the primary key (LHS must be a full key, unless RHS is a key)
- *No dependencies between non-prime attributes*

BCNF: A schema is in BCNF if (it is in 3NF and)

- LHS must be a super key
- *No dependencies between prime attributes*

Normalization - Finding Keys

Q5b) Consider the relation *R3 = (A, B, C, D)*, with the following functional dependencies:

- **AB -> C** and **C -> D**

What is the Candidate Key for this relation? What normal form does *R3* satisfy? You may assume that all tuples are unique and attributes are atomic.

Normalization - Finding Keys

Q5b) Consider the relation *R3 = (A, B, C, D)*, with the following functional dependencies:

- **AB -> C** and **C -> D**

What is the Candidate Key for this relation? What normal form does *R3* satisfy? You may assume that all tuples are unique and attributes are atomic.

Candidate Key is AB since:
AB -> C and
AB -> C -> D
so, with AB we can determine all attributes

Normal form is 2NF since C->D violates 3NF

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (A, B, D, E)**
- **R52 = (A, B, C)**

Will this be a loss-free decomposition, i.e., will we still be able to reconstruct all data by joining the two tables together? What normal form will *R51* and *R52* be in?

R5 = (<u>**A**</u>, <u>**B**</u>, C, <u>**D**</u>, E)

A -> C
BD -> C
ABD -> E

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (A, B, D, E)**
- **R52 = (A, B, C)**

Will this be a loss-free decomposition?

R5 = (A, B, C, D, E)

A -> C

BD -> C

ABD -> E

Lossless Decomposition test:

(from normalization lecture 2)

- **R1**, **R2** is a lossless join decomposition of **R** with respect to **F** **iff** at least one of the following dependencies is in **F+**
- **(R1 ∩ R2) → R1 – R2**
- **(R1 ∩ R2) → R2 – R1**

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (A, B, D, E)**
- **R52 = (A, B, C)**

Will this be a loss-free decomposition?

R5 = (<u>**A**</u>, <u>**B**</u>, C, <u>**D**</u>, E)

A -> C
BD -> C
ABD -> E

Lossless Decomposition test:

(from normalization lecture 2)

- **R1**, **R2** is a lossless join decomposition of **R** with respect to **F** **iff** at least one of the following dependencies is in **F+**
- **(R1 ∩ R2) → R1 – R2**
- **(R1 ∩ R2) → R2 – R1**

R51 ∩ R52 = AB
R51 - R52 = DE
R52 - R51 = C

AB -> C is part of F+

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (A, B, D, E)**
- **R52 = (A, B, C)**

R5 = (**A**, **B**, C, **D**, E)

A -> C
BD -> C
ABD -> E

What normal form will *R51* and *R52* be in?

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (**A, B, D, **E)**
- **R52 = (**A, B, **C)**

What normal form will *R51* and *R52* be in?

R5 = (**A**, **B**, C, **D**, E)

A -> C
BD -> C
ABD -> E

R51 is 3NF/BCNF since only ABD->E holds and ABD is the full candidate key

R52 is 1NF since A->C holds and A is a partial candidate key, so it cannot be 2NF

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- **R51 = (**A, B, D, **E)**
- **R52 = (**A, B, **C)**

How can we decompose and ensure 3NF for all relations?

R5 = (**A**, **B**, C, **D**, E)

A -> C
BD -> C
ABD -> E

Decomposition

Q6 Suppose we decompose Relation **R5** into two tables, **R51** and **R52**:

- R51 = (A, B, D, E)
- R52 = (A, B, C)

How can we decompose and ensure 3NF for all relations?

R5 = (**A**, **B**, C, **D**, E)

A -> C
BD -> C
ABD -> E

R51 is already 3NF

To fix R52 we could use
R53 = (A, C)

This must be 3NF

R51 ∩ R53 = A
R51 - R53 = BDE
R53 - R51 = C

A -> C is part of F+

Any other questions on Normalization?

Next: Shopping Cart

Shopping Cart Tips

Carefully read spec

- Make a list of tasks and workflows to test

Implement the tables from our ER diagram

Plan mockups of pages you will need

- Start with simplest requirements!
- Don’t worry about making it pretty until later

If your code won’t run… fix it!

- Don’t try to write a lot of code without testing