Warmup 1

1.
Reason1:View can simplify the user's effort to get the the data they are interested in by hiding the underlying querying and logic.
Reason2:View can restrict users' access to data by different level of authorization.

2.
Materialized view is fast, but it needs space to store the data.
Non-materialized view doesn't need extra space, but it's slower.

3.
For the first insertion to emp_names, there may be two possible results. The first is a new tuple inserted to the table employee where the dept_id, mgr_id and salary are all null or default value of that column. However, if there are some constrains for dept_id, mgr_id and salary columns, the insertion can't work.

The second insertion to avg_salaries can't work, because the view includes aggregation and grouping.

Warmup 2

1.

```
CHECK ( NOT EXISTS (
SELECT ID
FROM employee WHERE mgr_id IS NULL
EXCEPT
SELECT DISTINCT mgr_id
FROM employee))
```

2.
A lossy decomposition has lost meaningful information as a result of decomposition by adding noise
For example,

bname	bcity	assets	cname	lno	amt
Dtnn	Bkln	9M	Jones	L-17	1000
Dtnn	Bkln	9M	Johnson	L-23	2000
Mianus	Hnck	1.7M	Jones	L-93	500

If the relation decomposed to the two relations below, then this decomposition is lossy.

bname	bcity	assets	cname
Dtnn	Bkln	9M	Jones
Dtnn	Bkln	9M	Johnson
Mianus	Hnck	1.7M	Jones

bname	lno	amt
Dntn	L-17	1000
Dntn	L-23	2000
Mianus	L-93	500

To fix that, the relations above can be modified as below:

bname	assets	cname	lno
Dtnn	9M	Jones	L-17
Dtnn	9M	Johnson	L-23
Dtnn	9M	Jones	L-93

lno	bcity	amt
L-17	Bkln	1000
L-23	Bkln	2000
L-93	Hnck	500

3.
The dependency preservation decomposition is a property of decomposed relational database schema D in which each functional dependency $X \rightarrow Y$ specified in F either appeared directly in one of the relation schemas Ri in the decomposed D or could be inferred from the dependencies that appear in some Ri. Take the relation above for example, if $bname \rightarrow bcity\ assets$ and $lno \rightarrow amt\ bname$, then the following decomposition is not dependency preservation.

bname	assets	cname	lno
Dtnn	9M	Jones	L-17
Dtnn	9M	Johnson	L-23
Dtnn	9M	Jones	L-93

lno	bcity	amt
L-17	Bkln	1000
L-23	Bkln	2000
L-93	Hnck	500

To fix that, the relations above can be modified as below:

bname	bcity	assets	cname	lno
Dtnn	Bkln	9M	Jones	L-17
Dtnn	Bkln	9M	Johnson	L-23
Dtnn	Hnck	9M	Jones	L-93

lno	amt
L-17	1000
L-23	2000
L-93	500

4.
BCNF is not necessarily decomposition dependency preserving.

If (A,B,C,D) is decomposed to (A, B, D) and (B, C), then this BCNF decomposition is not dependency preserving, because $AB \rightarrow C$ is lost.

If (A,B,C,D) is decomposed to (A, B, C) and (A, D), then this BCNF decomposition is dependency preserving.

3NF is not necessarily decomposition dependency preserving, but it is always possible to find a dependency-preserving lossless-join decomposition that is in 3NF.

Problem 3

1.
The attributes that are not atomic:time_slot and evaluations.

Which attribute if any makes sense to split into
(a)multiple attributes
time_slot makes sense to split into multiple attributes because the time_slot is consisting of start_time and end_time.
evaluations is also OK to split into multiple attributes because we can use different attributes to indicate different kind of evaluation method.
(b)multiple tuples
Neither one
(c)a separate relation
evaluations makes sense to split into a separate relation if there are several kinds of evaluation method to make the relation more reasonable.

A question that can no longer be answered
What's the evaluation method of each of the course?

First Normal Form
Course (c_id, c_dept_id, c_dept, inst, office, sect, start_time, end_time)
Evaluation (evaluation_id, evaluation)
Link (c_id, evaluation_id)

2.
The course in not in seccond normal form, because there are some partial dependencies, such as $c\_id \rightarrow c\_dept\_id\ c\_dept$.

The only candidate key in the new course relation is c_id and section.

The attributes that depend on the entire candidate key:
inst office start_time end_time
The attributes that depend on the subset of the candidate key:
c_dept_id c_dep

Second Normal Form
Course(c_id, inst, office, sect, start_time, end_time)
Course_By_Dept (c_id, c_dept_id, c_dept)
Evaluation (evaluation_id, evaluation)
Link (c_id, evaluation_id)

3.
Axiom of transitivity

$c\_dept\_id \rightarrow\ c\_dept$ is a transitive dependency. c_dept_id is determined by c_id, however, it determines c_dept.

Third Normal Form
Course(c_id, inst, office, sect, start_time, end_time)
Course_By_Dept(c_id, c_dept_id)
Department(c_dept_id, c_dept)
Evaluation (evaluation_id, evaluation)
Link(c_id, evaluation_id)

4.
Explanation: If $\alpha \rightarrow \beta$ is trival, then
If α is superkey, then

$inst \rightarrow office$ does not satisfy BCNF. Because, instructor determines the office, but inst is not a superkey and this is not a trivial functional dependency.

BCNF
Course(c_id, inst, sect, start_time, end_time)
Office(inst, office)
Course_By_Dept(c_id, c_dept_id)
Department(c_dept_id, c_dept)
Evaluation (evaluation_id, evaluation)
Link(c_id, evaluation_id)

5.
(a)functional dependency: $c\_id\ sect \rightarrow ta$ and $ta \rightarrow sect$
(b) $c\_id\ sect \rightarrow ta$ ensures a course-section can only have one TA. $ta \rightarrow sect$ ensures each TA only teaches one section type.
(c)Yes, it's in 3NF. Because, there are no transitive dependencies.
(d)ta is not a super key, but $ta \rightarrow sect$.

(a)Table 4 is in BCNF. c_id and ta are candiate key together. Table 5 is in BCNF, because ta is a super key and the only functional dependency is $ta \rightarrow sect$
(b)$c\_id\ sect \rightarrow ta$ is lost
(c)Each course-section hires only one TA is not easy to guarantee.