1. (a)

```
select e.instructor_name, i.phone_number, e.department from
scheduling.employment as e join
scheduling.schedule as s on
e.instructor_name = s.instructor_name join
scheduling.instructor i on
s.instructor_name = i.instructor_name join
scheduling.classroom cr on
s.room_number = cr.room_number
where cr.room_number > 50;
```

1. (b)

```
select e.instructor_name, e.salary from scheduling.employment as e
where e.salary >
(select e.salary from scheduling.employment as e where e.instructor_name in ('Oliver Twist') and
e.department in ('CIS'));
```

1. (c)

```
select max(e.salary), min(e.salary), avg(e.salary), count(s.instructor_name) from
scheduling.employment as e join
scheduling.schedule as s on
e.instructor_name = s.instructor_name
where s.enrollment > 50;
```

1. (d)

```
select e.instructor_name, e.salary from
scheduling.employment as e join
scheduling.schedule as s on
e.instructor_name = s.instructor_name where
```

```
e.salary > (select e.salary from scheduling.employment as e join
scheduling.schedule as s on
e.instructor_name = s.instructor_name where
s.room_number in ('4211'));
```

2. (a)

```
WITH total_credit as (
select st.name, c.credits from
university.student as st join
university.gradereport as gr
on st.student_num = gr.student_num join
university.section as s on
s.section_id = gr.section_id join
university.course as c on
c.course_num = s.course_num
where st.name in (
Select st.name from
university.student as st join
university.gradereport as gr
on st.student_num = gr.student_num join
university.section as s on
s.section_id = gr.section_id
where s.course_num LIKE '%CIS%'
)) select distinct name, sum(credits) from total_credit group by name, credits;
```

2. (b)

```
select s.semester, s.year, st.name
from university.student as st join
university.gradereport as gr
```

```
on st.student_num = gr.student_num join
university.section as s on
s.section_id = gr.section_id join
university.course as c on
c.course_num = s.course_num where
s.course_num IN ("CIS 9340", "CIS 9467")
group by st.name
having count(distinct s.year) = 1 AND count(distinct s.semester) = 1;
```

3.

As the most recent backup is available, the DBA may re-preform the system recovery process. That is, do it again. A trigger can be written wherein, whenever a commit fails halfway or throws an error before full execution, it must restore the last saved image of the database.

Once the last image is restored, we have basically reverted ourselves back to the place where it was before the previous command was executed. The failure or success of it will not matter anymore as we have re-stored it to the last instance that was in working state.

An example for restoring any database back to the previous instance is done as follows:

```
USE master

RESTORE DATABASE <database name>
FROM DISK = 'C:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\
MSSQL\Backup\<backup name>.bak'
WITH FILE = 3
        ,REPLACE
        ,NORECOVERY;

RESTORE <instance name>
```

```
FROM DISK = 'D:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\
MSSQL\Backup\<backup name>.bak'

RESTORE LOG <database name>
FROM DISK = 'D:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\
MSSQL\Backup\<backup name>.bak'
WITH FILE = 4 ,NORECOVERY ,STOPAT = <timestamp ip-address>';

RESTORE LOG <database name>
FROM DISK = 'D:\Program Files\Microsoft SQL Server\MSSQL11.MSSQLSERVER\
MSSQL\Backup\<backup name>.bak'
WITH FILE = 5 ,NORECOVERY ,STOPAT = timestamp ip-address;
```