DB REV TUT4

1. A new government initiative to get more young people into work cuts the salary levels of all workers under 25 by 20%. Write an SQL statement to implement this policy change.
    1. the question is asking: reduce all young worker's salary by 20%
    2. answer sql: using update keyword

```sql
update Employees
set salary = salary * 0.8
where age < 25
```

~~~
2.  Write an SQL statement to give all employees in the Sales department a
10% pay rise.
    1. question is asking: give sales employees a 10% pay rise
    2. answer sql:
        1. Find department which dname = 'sales' --> take did
        2. Find WorksIn which did = department.did --> take eid
        3. For Employees which have this eid, update salary
     ```sql
     update Employees e
     set salary = salary * 1.1
     where e.eid in (
         select w.eid
         from worksIn w, department d
         where d.d.name = 'Sales' and d.did = w.did
     )
~~~

3. Add a constraint to the CREATE TABLE statements above to ensure that every department must have a manager.
    1. make manager field in departments table not null

```sql
-- before changes:
create table Departments (
    did integer,
    ...
    manager integer references Employees(eid)
)
```

```sql
-- after changes:
create table Departments (
    did integer,
    ...
    manager integer not null,
    foreign key (manager) references Employees(eid)
)
```

~~~
4. Add a constraint to the CREATE TABLE statements above to ensure that no-
one is paid less than the minimum wage of $15,000.
    1. make salary in Employees table more than 15000
    ```sql
    -- before changes:
    create table Employees (
        salary real,
        ...
    )
    -- after changes:
    create table Employees (
        salary real check (salary >= 15000) -- add check statement --
    )
~~~

5. When an employee is removed from the database, it makes sense to also delete all of the records that show which departments he/she works for. Modify the CREATE TABLE statements above to ensure that this occurs.
    1. the question is asking: when delete one employee, also delete all of the records (works in table)

```sql
-- use CASCADE
-- CASCADE refers to an action triggered automatically when a change
occurs in a parent table
-- usage:
    -- ON DELETE CASCADE
    -- ON UPDATE CASCADE
-- purpose: ensure referenial integrity, maintain consistent
relationships, reduce the complexity of manual database management tasks
create table WorksIn (
    eid integer,
    did integer,
    percent real,
    primary key (eid, did)
```

```
    foreign key (eid) references Employees(eid) ON DELETE CASCADE,
    foreign key (did) references Departments(did)
)
```

~~~
6. Find the _names_ of suppliers who supply some red part.
    1. find parts with p.color = 'red' --> pid
    2. find catalog with pid = p.pid --> sid
    3. find suppliers with s.sid = sid --> names
    ```sql
    select s.names
    from suppliers s
    join parts p, catalog c
    where p.color = 'read' and c.pid = p.pid and s.sid = c.sid
~~~

7. Find the sids of suppliers who supply some red or green part.
    1. find parts with parts.color = red or green --> pid
    2. find catalog with c.pid = p.pid --> sid

```sql
select c.sid
from catalog c where c.pid = p.pid
join parts p where p.color = 'red' or p.color = 'green'
```

~~~
8. Find the _sids_ of suppliers who supply some red part or whose address is
221 Packer Street.
    1. red part:
        1. find parts with parts.color = red --> pid
        2. find catalog with c.pid = p.pid --> sid
    2. address
        1. find suppliers with s.address = "221 packer street"
    ```sql
    select s.sid
    from supplier s
    where s.address = "221 packer street"
        or s.sid in (
                select c.sid
                from parts p, catalog c
~~~

```
                where p.color = 'red' and c.pid = p.pid
        )
```

9. Find the *sids* of suppliers who supply some red part and some green part.
    1. find parts where p.color = red --> pid
    2. find catalog where c.pid = p.pid --> sid
    3. then check for this sid, if also exist "red parts"

```sql
select c.sid
from parts p, catalogs c
where p.color = 'red' and c.pid = p.pid
    and exist (
        select c2.sid
        from parts p2, catalogs c2
        where p2.color = 'green' and c2.pid = p2.pid and c2.sid = c.sid
    )
```

~~~
10. Find the _sids_ of suppliers who supply every part.
    1. grab all the parts id (setA)
    2. grab all the catalog where this supplier did --> pid (setB)
    3. find all supplier sid where setA - setB = 0
    4. example:
        1. supplier: 1, 2
        2. parts: A., B
        3. catalog: 1-A, 1-B, 2-A
        4. run algo:
            1. first try: sid = 1
                1. setA = A, B
                2. setB = A, B
                3. setA - setB = 0, check success, 1 is what we want
            2. second try: sid = 2
                1. setA = A, B
                2. setB = A
                3. setA - setB != 0, check failed, 2 is not what we want
    ```sql
    select s.sid
    from supplier s
    where not exist (
        (select pid from parts)
~~~

```
        except
        (
            select c.pid
            from catalog c
            where c.sid = s.sid
        )
    )
```

11. Find the *sids* of suppliers who supply every red part.

    ```sql
    select s.sid
    from supplier s
    where not exists (
        (select pid from parts where color = 'red')
        except
        (
            select c.pid from catalog c where c.sid = s.sid
        )
    )
    ```

~~~
12. Find the _sids_ of suppliers who supply every red or green part.
    ```sql
    select s.sid
    from supplier s
    where not exists (
        (select pid from parts where color = 'red' or color = 'green')
        except
        (
            select c.pid from catalog c where c.sid = s.sid
        )
    )
~~~

13. Find the *sids* of suppliers who supply every red part or supply every green part.
    1. idea:
        1. find suppliers who supply every red part --> setA
        2. find suppliers who supply every green part --> setB
        3. setA union setB

```sql
(
    select s.sid
    from supplier s
    where not exists (
        (select pid from parts where color = 'red')
        except
        (select c.pid from catalog c where c.sid = s.sid)
    )
)
union
(
    select s.sid
    from supplier s
    where not exists (
        (select pid from parts where color = 'red')
        except
        (select c.pid from catalog c where c.sid = s.sid)
    )
)
```

~~~
14. Find pairs of _sids_ such that the supplier with the first _sid_ charges
more for some part than the supplier with the second _sid_.
    1. make two column,
    2. take info from catalog table
    3. first.sid != second.sid
    4. first.cost > second.cost
    5. first.pid = second.pid
    ```sql
    select c1.sid, c2.sid
    from catalogs c1, catalogs c2
    where c1.sid != c2.sid and c1.pid = c2.pid and c1.cost > c2.cost
~~~

15. Find the *pids* of parts that are supplied by at least two different suppliers.
    1. use catalog table
    2. lock a pid, then check if exist for the same pid in the catalog table, there is another entry

```sql
select c.pid
from catalogs c
where exists (
    select c1.pid
```

```
    from catalogs c1
    where c1.pid = c.pid and c1.sid != c.sid
)
```

~~~
16. Find the _pids_ of the most expensive part(s) supplied by suppliers
named "Yosemite Sham".
    1. supplier name = "yosemite sham" --> sid
    2. catalog sid = s.sid --> cost
    3. then find cost >= all cost
    ```sql
    select c.pid
    from catalog c, suppliers s
    where s.name = "yosemite sham" and c.sid = s.sid
        and c.cost >= all(
            select c1.cost
            from catalog c1, suppliers s1
            where s1.name = "yosemite sham" and c1.sid = s1.sid
        )
~~~

17. Find the *pids* of parts supplied by every supplier at a price less than 200 dollars (if any supplier either does not supply the part or charges more than 200 dollars for it, the part should not be selected).
    1. from catalog table: catalog.cost <= 200 --> pid
    2. **and need to ensure that EVERY supplier have 200$ less price**

```sql
select c.pid
from catalog c
where c.cost <= 200
group by c.sid
having count(*) = (select count(*) from suppliers) -- EVERY supplier
```