CS 145 – PSET 1 – Sample Answers – 2016

NOTE: there are more possible answers than those listed here

1a

```sql
SELECT a.i,
       a.j,
       a.val * b.val AS val
FROM   a,
       b
WHERE  a.i = b.i
       AND a.j = b.j;
```

```sql
SELECT a.i,
       a.j,
       a.val * b.val
FROM   a,
       b
WHERE  a.i = b.i
       AND a.j = b.j;
```

```sql
SELECT a.i           AS "i",
       a.j           AS "j",
       a.val * b.val AS "val"
FROM   a,
       b
WHERE  a.i = b.i
       AND a.j = b.j;
```

1b

```sql
SELECT c.pi AS i,
       a.j,
       a.val
FROM   a,
       c
WHERE  a.i = c.ind
ORDER  BY c.pi;
```

```sql
SELECT c.pi AS i,
       a.j,
       a.val
FROM   a,
       c
WHERE  a.i = c.ind
ORDER  BY i,
          j;
```

```sql
SELECT c.pi AS "i",
       a.j   AS "j",
       a.val AS "val"
FROM   a,
       c
WHERE  a.i = c.ind
ORDER  BY c.pi;
```

1c

```sql
SELECT comp.pi AS i,
       a.j,
       a.val
FROM   a,
       (SELECT c1.ind,
               c2.pi
        FROM   c c1,
               c c2
        WHERE  c1.pi = c2.ind) AS comp
WHERE  a.i = comp.ind
ORDER  BY comp.pi;
```

```
SELECT compose.pi AS i,
       a.j,
       a.val
FROM   a,
       (SELECT c1.ind,
               c2.pi
        FROM   c AS c1,
               c AS c2
        WHERE  c1.pi = c2.ind) compose
WHERE  a.i = compose.ind
ORDER  BY i,
          j;
```

```
SELECT c2.pi AS "i",
       a.j    AS "j",
       a.val  AS "val"
FROM   a,
       c AS c1,
       c AS c2
WHERE  a.i = c1.ind
       AND c1.pi = c2.ind
ORDER  BY c2.pi;
```

1d

```
SELECT a1.i,
       a1.j,
       Max(a2.val) AS val
FROM   a a1,
       a a2
WHERE  Abs(a1.i-a2.i) + Abs(a1.j-a2.j) <= 1
GROUP  BY a1.i,
          a1.j;
```

```
SELECT A1.i,
       A1.j,
       Max(A2.val) AS val
FROM   a AS A1,
       a AS A2
WHERE  Abs(A1.i - A2.i) + Abs(A1.j - A2.j) <= 1
GROUP  BY A1.i,
          A1.j;
```

```
SELECT origin.i            AS i,
       origin.j            AS j,
       Max(candidates.val) AS val
FROM   a AS origin,
       a AS candidates
WHERE  ( origin.i = candidates.i
         AND ( origin.j = candidates.j
                OR origin.j = candidates.j + 1
                OR origin.j = candidates.j - 1 ) )
        OR ( origin.j = candidates.j
             AND ( origin.i = candidates.i
                    OR origin.i = candidates.i + 1
                    OR origin.i = candidates.i - 1 ) )
GROUP  BY origin.i,
          origin.j;
```

2a

```sql
SELECT region,
       Count(region) AS num_state_winners
FROM   energy,
       (SELECT Max(solar)   AS maxsolar,
               Max(wind)    AS maxwind,
               Max(hydro)   AS maxhydro,
               Max(nuclear) AS maxnuclear
        FROM   energy)
WHERE  solar = maxsolar
        OR wind = maxwind
        OR hydro = maxhydro
        OR nuclear = maxnuclear
GROUP  BY region
ORDER  BY num_state_winners DESC;
```

```sql
SELECT region,
       Count(state) num_state_winners
FROM   energy,
       (SELECT Max(solar)   ms,
               Max(wind)    mw,
               Max(hydro)   mh,
               Max(nuclear) mn
        FROM   energy) maxes
WHERE  energy.solar = ms
        OR energy.wind = mw
        OR energy.hydro = mh
        OR energy.nuclear = mn
GROUP  BY region
ORDER  BY num_state_winners DESC;
```

```sql
SELECT e1.region       AS region,
       Count(e1.state) AS num_state_winners
FROM   energy e1
WHERE  ( e1.solar = (SELECT Max(e2.solar)
                     FROM   energy e2) )
        OR ( e1.wind = (SELECT Max(e3.wind)
                        FROM   energy e3) )
        OR ( e1.hydro = (SELECT Max(e3.hydro)
                         FROM   energy e3) )
        OR ( e1.nuclear = (SELECT Max(e4.nuclear)
                           FROM   energy e4) )
GROUP  BY e1.region
ORDER  BY num_state_winners DESC;
```

2b

```sql
SELECT state,
       solar,
       wind
FROM   (SELECT state,
               solar,
               wind,
               ( solar + wind ) AS sum
        FROM   energy AS e1
        WHERE  NOT EXISTS (SELECT e2.solar,
                                  e2.wind
                           FROM   energy AS e2
                           WHERE  e2.solar >= e1.solar
                                  AND e2.wind > e1.wind)
               AND NOT EXISTS (SELECT e2.solar,
                                      e2.wind
                               FROM   energy AS e2
                               WHERE  e2.solar > e1.solar
                                      AND e2.wind >= e1.wind))
ORDER  BY sum DESC;
```

```
SELECT state,
       solar,
       wind
FROM   energy
WHERE  state NOT IN (SELECT e1.state
                     FROM   energy e1,
                            energy e2
                     WHERE  ( ( e2.solar >= e1.solar
                                AND e2.wind > e1.wind )
                               OR ( e2.solar > e1.solar
                                    AND e2.wind >= e1.wind ) ))
ORDER  BY solar + wind DESC;
```

```
SELECT e1.state,
       e1.solar,
       e1.wind
FROM   energy e1,
       energy e2
WHERE  NOT EXISTS(SELECT *
                  FROM   energy e2
                  WHERE  e1.state <> e2.state
                         AND e2.wind >= e1.wind
                         AND e2.solar >= e1.solar)
GROUP  BY e1.state
ORDER  BY ( e1.solar + e1.wind ) DESC;
```

2c

```
SELECT n1.region
FROM   (SELECT region,
               Min(nuclear) AS min_state
        FROM   energy
        WHERE  nuclear > 0.0
        GROUP  BY region) AS n1
WHERE  n1.min_state >= (SELECT 0.1 * Max(n2.nuclear)
                        FROM   energy AS n2);
```

```
SELECT region
FROM   energy,
       (SELECT Max(nuclear) max_nuke
        FROM   energy) mn
WHERE  nuclear > 0.0
GROUP  BY region
HAVING Min(nuclear) >= 0.1 * max_nuke;
```

```
SELECT e1.region
FROM   energy e1
WHERE  (SELECT Min(e2.nuclear)
        FROM   energy e2
        WHERE  e1.region = e2.region
               AND e2.nuclear > 0) > ( 0.1 * (SELECT Max(e3.nuclear)
                                              FROM   energy e3) )
GROUP  BY region;
```

2d

```
SELECT region
FROM   (SELECT region,
               nuclear AS min_state
        FROM   energy AS e1
        WHERE  nuclear > 0.0
               AND NOT EXISTS (SELECT e2.nuclear
                               FROM   energy AS e2
                               WHERE  e2.region = e1.region
```

```
                                      AND nuclear > 0.0
                                      AND e2.nuclear < e1.nuclear)) AS e3
WHERE  NOT EXISTS (SELECT e4.nuclear
                   FROM   energy AS e4
                   WHERE  0.1 * e4.nuclear > e3.min_state);
```

```
SELECT e.region
FROM   energy e,
       (SELECT e1.nuclear max_nuke
        FROM   energy e1
        WHERE  NOT EXISTS (SELECT e2.nuclear
                           FROM   energy e2
                           WHERE  e2.nuclear > e1.nuclear))
WHERE  NOT EXISTS (SELECT e3.nuclear
                   FROM   energy e3
                   WHERE  e.region = e3.region
                          AND e3.nuclear > 0.0
                          AND e3.nuclear < e.nuclear)
       AND e.nuclear > 0.1 * max_nuke;
```

```
SELECT e1.region
FROM   energy e1
WHERE  e1.nuclear > 0
       AND NOT EXISTS (SELECT *
                       FROM   energy e2
                       WHERE  e2.region = e1.region
                              AND e2.nuclear < e1.nuclear
                              AND e1.state <> e2.state
                              AND e2.nuclear > 0)
       AND NOT EXISTS(SELECT *
                      FROM   energy e3
                      WHERE  ( e3.nuclear * 0.1 ) > e1.nuclear
                             AND e1.state <> e3.state);
```

3a

```
SELECT x.i,
       Sum(x.val * model.val) AS val
FROM   x,
       model
WHERE  x.j = model.j
GROUP  BY x.i
LIMIT  5;
```

```
SELECT i,
       Sum(x.val * model.val) val
FROM   x,
       model
WHERE  x.j = model.j
GROUP  BY i
LIMIT  5;
```

```
SELECT x.i                    AS "i",
       Sum(x.val * model.val) AS "val"
FROM   x,
       model
WHERE  x.j = model.j
GROUP  BY x.i
LIMIT  5;
```

3b

```
DROP VIEW IF EXISTS prediction;

CREATE VIEW prediction
AS
  SELECT A.i,
         ( Sum(A.val * B.val) >= 0 ) AS val
  FROM   x AS A,
         model AS B
  WHERE  A.j = B.j
  GROUP  BY A.i;
```

```
DROP VIEW IF EXISTS prediction;

CREATE VIEW prediction
AS
  SELECT i,
         val > 0 val
  FROM   answer_p3a;
```

```
DROP VIEW IF EXISTS prediction;

CREATE VIEW prediction
AS
  SELECT x.i                          AS "i",
         Sum(x.val * model.val) > 0 AS "val"
  FROM   x,
         model
  WHERE  x.j = model.j
  GROUP  BY x.i;
```

3c

```
SELECT Avg(p.val = iris.label) AS accuracy
FROM   prediction p,
       iris
WHERE  p.i = iris.i;
```

```
SELECT Cast(correct AS FLOAT) / Count(i) accuracy
FROM   iris,
       (SELECT Count() correct
        FROM   answer_p3b prediction,
               iris
        WHERE  prediction.i = iris.i
               AND prediction.val = iris.label);
```

```
SELECT Avg(p.val = iris.label) AS "accuracy"
FROM   prediction p,
       iris
WHERE  p.i = iris.i;
```

Bonus

```
SELECT model.j,
       model.val + Sum(DELTA.val*ie.val) AS val
FROM   model,
       x ie,
       (SELECT iris.i,
               0.0001 * ( iris.label - p.val ) AS val
        FROM   iris,
               prediction p
        WHERE  iris.i = p.i) AS DELTA
WHERE  ie.i = DELTA.i
       AND ie.j = model.j
GROUP  BY model.j;
```

```
SELECT model.j,
       model.val + deviation
FROM   model,
       (SELECT x.j,
               0.0001 * Sum(x.val * corr.diff) deviation
        FROM   x,
               (SELECT iris.i,
                       ( iris.label - prediction.val ) diff
                FROM   iris,
                       answer_p3b prediction
                WHERE  iris.i = prediction.i) corr
        WHERE  x.i = corr.i
        GROUP  BY x.j) upd
WHERE  model.j = upd.j;
```

```
DROP view IF EXISTS next_step;

CREATE view next_step
AS
  SELECT 0                                                       AS j,
         0.0001 * Sum(( data.label - p.val ) * data.sepallength) AS "val"
  FROM   prediction p,
         iris data
  WHERE  data.i = p.i
  UNION
  SELECT 1                                                      AS j,
         0.0001 * Sum(( data.label - p.val ) * data.sepalwidth) AS "val"
  FROM   prediction p,
         iris data
  WHERE  data.i = p.i
  UNION
  SELECT 2                                                       AS j,
         0.0001 * Sum(( data.label - p.val ) * data.petallength) AS "val"
  FROM   prediction p,
         iris data
  WHERE  data.i = p.i
  UNION
  SELECT 3                                                      AS j,
         0.0001 * Sum(( data.label - p.val ) * data.petalwidth) AS "val"
  FROM   prediction p,
         iris data
  WHERE  data.i = p.i;

SELECT w.j AS "j",
       w.val + s.val
FROM   model w,
       next_step s
WHERE  w.j = s.j;
```