1.

List a company's workers by names.

```
SELECT per_id,
per_name

FROM person
NATURAL JOIN works
NATURAL JOIN job

WHERE comp_id = '8'
AND end_date IS NULL;
```

2

List a company's staff by salary in descending order.

```
SELECT per_name,
pay_rate

FROM person
NATURAL JOIN works
NATURAL JOIN job

WHERE comp_id = '8'
AND pay_type = 'salary'

ORDER BY pay_rate DESC;
```

3

List companies' labor cost (total salaries and wage rates by 1920 hours) in descending order.

```
SELECT comp_id,
2
         SUM (CASE
3
               WHEN pay_type = 'salary' THEN pay_rate
4
                ELSE pay_rate * 1920
5
              END) AS total_labor_cost
6
  FROM
         job
         NATURAL JOIN works
8
  GROUP BY comp_id
  ORDER BY total_labor_cost DESC;
```

4

Find all the jobs a person is currently holding and worked in the past.

```
SELECT job_code,
start_date,
end_date

FROM works
NATURAL JOIN job

WHERE per_id = 1

ORDER BY start_date DESC;
```

5

List a person's knowledge/skills in a readable format.

```
SELECT ks_title,

ks_level,

ks_description

FROM has_skill

NATURAL JOIN knowledge_skill

WHERE per_id = 1;
```

6

List the skill gap of a worker between his/her job(s) and his/her skills.

```
(SELECT ks_code
2
   FROM
          required_skill
3
          NATURAL JOIN works
4
   WHERE per_id = 1)
5
  MINUS
6
  (SELECT ks_code
7
          has_skill
   FROM
   WHERE per_id = 1);
```

7

List the required knowledge/skills of a job/ a job category in a readable format. (two queries)

```
2
   -- a job
3
   SELECT ks_code,
4
          ks_title,
5
          ks_level,
6
          ks_description
7
          required_skill
   FROM
8
          NATURAL JOIN knowledge_skill
9
   WHERE job_code = 1; ;
   -- a job category
10
11
   SELECT ks_code,
          ks_title,
12
          ks_level,
13
14
          ks_description
15
   FROM
          skill_set
          NATURAL JOIN knowledge_skill
16
   WHERE cate_code = 1;
```

8

List a person's missing knowledge/skills for a specific job in a readable format.

```
(SELECT ks_code,
2
            ks_title
3
    FROM
            required_skill
            NATURAL JOIN knowledge_skill
4
5
    WHERE
           job\_code = 2)
6
   MINUS
7
   (SELECT ks_code,
8
            ks_title
            has_skill
9
    FROM
10
           NATURAL JOIN knowledge_skill
    WHERE per_id = 1);
```

9

List the courses (course id and title) that each alone teaches all the missing knowledge/skills for a person to pursue a specific job.

```
1
```

```
WITH missing_ks(ks)
3
        AS ((SELECT ks_code
4
                      required_skill
              FROM
5
              WHERE
                     job\_code = 1)
6
             MINUS
7
             (SELECT ks_code
8
              FROM
                      has_skill
9
              WHERE per_id = 1))
   SELECT c_code,
11
           c_title
   FROM
           course c
12
   WHERE NOT EXISTS((SELECT *
13
14
                        FROM
                               missing_ks)
15
                       MINUS
16
                       (SELECT ks_code
17
                        FROM
                               teaches_skill ts
18
                        WHERE
                               ts.c_code = c.c_code));
```

10

Suppose the skill gap of a worker and the requirement of a desired job can be covered by one course. Find the "quickest" solution for this worker. Show the course, section information and the completion date.

```
WITH missing_ks(ks)
2
        AS ((SELECT ks_code
3
              FROM
                     required_skill
                     job_code = 1)
4
              WHERE
5
             MINUS
6
             (SELECT ks_code
7
              FROM
                     has_skill
              WHERE per_id = 1)),
8
        fulfilling_courses(c_code)
9
10
        AS (SELECT c_code
11
             FROM
                    course c
             WHERE NOT EXISTS ((SELECT *
12
13
                                  FROM
                                          missing_ks)
14
                                 MINUS
15
                                 (SELECT ks_code
                                          teaches_skill ts
16
                                  FROM
17
                                  WHERE ts.c_code = c.c_code))),
        fulfilling_section(c_code, complete_date)
18
```

```
AS (SELECT DISTINCT c_code,
19
20
                              complete_date
21
             FROM
                    SECTION
                    NATURAL JOIN fulfilling_courses
23
                    complete_date >= Trunc(SYSDATE))
             WHERE
24
   SELECT c_code,
25
           complete_date
           fulfilling_section
26
   FROM
   WHERE complete_date = (SELECT Min(complete_date)
27
28
                             FROM
                                    fulfilling_section);
```

11

Find the cheapest course to make up one's skill gap by showing the course to take and the cost (of the section price).

```
WITH missing_ks(ks)
2
        AS ((SELECT ks_code
3
                     required_skill
              FROM
4
              WHERE
                     job\_code = 1)
5
             MINUS
6
             (SELECT ks_code
7
                     has_skill
              FROM
              WHERE per_id = 1)),
8
         fulfilling_courses(c_code, c_title, retail_price)
9
10
        AS (SELECT c_code,
11
                    c_title,
12
                    retail_price
13
             FROM
                    course c
14
                    NOT EXISTS ((SELECT *
             WHERE
15
                                   FROM
                                          missing_ks)
16
                                 MINUS
                                  (SELECT ks_code
18
                                  FROM
                                          teaches_skill ts
19
                                  WHERE ts.c_code = c.c_code)))
20
   SELECT c_code,
21
           c_title,
22
           retail_price
23
   FROM
          fulfilling_courses
24
   WHERE retail_price = (SELECT Min(retail_price)
25
                                    fulfilling_courses
                            FROM
26
                                    NATURAL JOIN SECTION);
```

12

If query #9 returns nothing, then find the course sets that their combination covers all the missing knowledge/ skills for a person to pursue a specific job. The considered course sets will not include more than three courses. If multiple course sets are found, list the course sets (with their course IDs) in the order of the ascending order of the course sets' total costs.

13

List all the job categories that a person is qualified for.

```
2
   SELECT cate_code,
3
          cate_title
4
   FROM
          job_category jc
          NOT EXISTS ((SELECT ks_code
5
   WHERE
6
                         FROM
                                skill_set ss
7
                         WHERE jc.cate_code = ss.cate_code)
8
                        MINUS
9
                        (SELECT ks_code
10
                         FROM
                                has_skill
                        WHERE per_id = 2));
11
```

14

Find the job with the highest pay rate for a person according to his/her skill qualification

```
WITH qualified_jobs
2
        AS (SELECT j.job_code
3
             FROM
                    job j
             WHERE
                    NOT EXISTS ((SELECT ks_code
4
5
                                          required_skill rs
                                  FROM
                                  WHERE j.job_code = rs.job_code)
6
7
                                 MINUS
8
                                 (SELECT ks_code
9
                                  FROM
                                          has_skill
                                  WHERE per_id = 1))),
10
        q_jobs_desc
11
```

```
12
        AS (SELECT *
13
             FROM
14
                     NATURAL JOIN qualified_jobs)
15
   SELECT job_code,
16
           pay_rate,
17
           pay_type
18
   FROM
           q_jobs_desc
19
   WHERE pay_rate = (SELECT Max(CASE
                                      WHEN pay_type = 'salary' THEN pay_rate
20
21
                                      ELSE pay_rate * 1920
22
                                    END)
23
                                q_jobs_desc);
                        FROM
```

15

List all the names along with the emails of the persons who are qualified for a job.

```
SELECT per_name,
2
           email
3
   FROM
          person p
4
   WHERE
          NOT EXISTS ((SELECT ks_code
5
                        FROM
                                required_skill
6
                        WHERE job_code = 1)
7
                       MINUS
8
                        (SELECT ks_code
9
                                has_skill hs
                        FROM
10
                        WHERE hs.per_id = p.per_id));
```

16

When a company cannot find any qualified person for a job, a secondary solution is to find a person who is almost qualified to the job. Make a "missing-one" list that lists people who miss only one skill for a specified job.

```
SELECT per_id,
per_name
FROM person p
WHERE 1 = (SELECT Count(ks_code)
FROM ((SELECT ks_code
FROM required_skill
```

```
WHERE job_code = 1)

MINUS

(SELECT ks_code

FROM has_skill hs

WHERE hs.per_id = p.per_id)));
```

17

List the skillID and the number of people in the missing-one list for a given job code in the ascending order of the people counts.

```
WITH skills_needed(ks_code)
2
        AS (SELECT ks_code
             FROM
                    required_skill
4
             WHERE job_code = '1'),
5
        missing_skills(per_id, ms_count)
6
        AS (SELECT per_id,
7
                    Count(ks_code)
8
             FROM
                    person p,
9
                    skills_needed
                    ks_code IN ((SELECT ks_code
10
             WHERE
11
                                  FROM
                                          skills_needed)
12
                                 MINUS
13
                                 (SELECT ks_code
14
                                  FROM
                                          has_skill
15
                                  WHERE per_id = p.per_id))
16
             GROUP BY per_id)
   SELECT ks_code,
17
           Count(per_id) AS total_ms_count
18
          missing_skills ms,
   FROM
19
20
           skills_needed
21
   WHERE
          ks_code IN ((SELECT ks_code
22
                                skills_needed)
                         FROM
23
                       MINUS
24
                        (SELECT ks_code
25
                                has_skill
                         FROM
26
                        WHERE per_id = ms.per_id))
27
          AND ms_count = 1
   GROUP BY ks_code
28
   ORDER BY total_ms_count ASC;
```

18

Suppose there is a new job that has nobody qualified. List the persons who miss the least number of skills and report the "least number".

```
WITH skills_needed(ks_code)
2
        AS (SELECT ks_code
3
                    required_skill
             FROM
4
             WHERE job_code = 1),
        missing_skills(per_id, ms_count)
5
6
        AS (SELECT per_id,
                    Count(ks_code)
7
8
             FROM
                    person p,
9
                    skills_needed sn
10
                    sn.ks_code IN ((SELECT ks_code
             WHERE
                                             required_skill)
11
                                      FROM
                                     MINUS
12
13
                                     (SELECT ks_code
14
                                      FROM
                                             has_skill
15
                                      WHERE
                                             per_id = p.per_id))
16
             GROUP BY per_id),
17
        min_missing_ks(min_ms_count)
18
        AS (SELECT Min(ms_count)
19
             FROM
                    missing_skills)
   SELECT per_id,
21
          ms_count
   FROM
          missing_skills
           JOIN min_missing_ks
24
             ON ms_count = min_missing_ks.min_ms_count;
```

19

For a specified job category and a given small number k, make a "missing-k" list that lists the people's IDs and the number of missing skills for the people who miss only up to k skills in the ascending order of missing skills.

```
WITH skills_needed(ks_code)

AS (SELECT ks_code

FROM required_skill

WHERE job_code = 1),
```

```
missing_skills(per_id, ms_count)
6
7
        AS (SELECT per_id,
8
                    Count(ks_code)
9
             FROM
                    person p,
                     (SELECT ks_code
11
                      FROM
                             skills_needed) sn
12
                    sn.ks_code IN ((SELECT ks_code
                                             skills_needed)
13
                                      FROM
                                     MINUS
14
15
                                     (SELECT ks_code
                                      FROM
                                             has_skill
16
17
                                      WHERE per_id = p.per_id))
18
             GROUP
                    BY per_id)
   SELECT per_id,
19
20
           ms_count
   FROM
           missing_skills
21
   WHERE ms_count <= 3 --k
22
23
   ORDER BY ms_count ASC;
```

20

Given a job category code and its corresponding missing-k list specified in Question 19. Find every skill that is needed by at least one person in the given missing-k list. List each skillID and the number of people who need it in the descending order of the people counts.

```
WITH skills_needed(ks_code)
2
        AS (SELECT ks_code
3
                    required_skill
             FROM
                   job\_code = '1'),
4
             WHERE
5
        missing_skills(per_id, ms_count)
        AS (SELECT per_id,
6
7
                    Count(ks_code)
8
             FROM
                    person p,
9
                    (SELECT ks_code
10
                             skills_needed) sn
                    sn.ks_code IN ((SELECT ks_code
11
             WHERE
                                             skills_needed)
12
                                      FROM
13
                                     MINUS
                                     (SELECT ks_code
14
15
                                      FROM
                                             has_skill
16
                                             per_id = p.per_id))
                                      WHERE
17
             GROUP
                   BY per_id),
```

```
missing_people(per_id, ms_count)
18
19
        AS (SELECT per_id,
20
                    ms_count
                    missing_skills
21
             FROM
22
             WHERE ms_count <= 3)</pre>
23
   SELECT ks_code,
           Count(per_id) AS mp_count
24
25
   FROM
           missing_people p,
           skills_needed
26
   WHERE skills_needed.ks_code IN (SELECT ks_code
27
28
                                              skills_needed
                                       FROM
29
                                       MINUS
                                       SELECT ks_code
                                              has_skill
31
                                       FROM
                                       WHERE per_id = P.per_id)
32
   GROUP BY ks_code
   ORDER BY mp_count DESC;
```

21

In a local or national crisis, we need to find all the people who once held a job of the special job category identifier.

```
SELECT per_id
FROM works NATURAL JOIN job NATURAL JOIN job_category
where cate_code = 1;
```

22

Find all the unemployed people who once held a job of the given job identifier.

```
WITH unemployed(per_id)
2
        AS ((SELECT per_id
3
              FROM
                     person)
4
             MINUS
5
             (SELECT per_id
6
              FROM
                     works
7
              WHERE end_date >= current_date))
8
   SELECT per_id
9
          unemployed
           NATURAL JOIN works
10
```

```
WHERE job_code = 8;
```

23

Find out the biggest employer in terms of number of employees or the total amount of salaries and wages paid to employees.

```
WITH company_size(comp_id, employee_count)
2
       AS (SELECT comp_id,
3
                   Count(*)
4
            FROM
                   job
5
                   NATURAL JOIN works
6
            GROUP
                   BY comp_id)
7
  SELECT comp_id employee_COUNT
8
          company_size
  FROM
9
  WHERE employee_count = (SELECT Max (employee_count)
                             FROM
                                    company_size);
```

24

Find out the job distribution among business sectors; find out the biggest sector in terms of number of employees or the total amount of salaries and wages paid to employees.

```
WITH sector_size(primary_sector, employee_count)
2
        AS (SELECT primary_sector,
3
                    Count(*)
4
             FROM
                    job
5
                    NATURAL JOIN works
6
                    NATURAL JOIN company
            GROUP BY primary_sector)
   SELECT primary_sector,
8
9
          employee_count
10
          sector_size
   FROM
11
   WHERE
          employee_count = (SELECT Max (employee_count)
12
                                     sector_size);
                              FROM
```

- 25. Find out the ratio between the people whose earnings increase and those – whose earning decrease; find the average rate of earning improvement for the – workers in a specific business sector. – this does not work – did not do.

```
--WITH
--pay_rate_from_work AS (
--SELECT per_id, works.job_code, start_date, end_date, case
-- when pay_type = '
salary'

-- then pay_rate
else pay_rate*1920
end
-- end
```

- 26. Find the leaf-node job categories that have the most openings due to − lack of qualified workers. If there are many opening jobs of a job category − but at the same time there are many qualified jobless people. Then training − cannot help fill up this type of job. What we want to find is such a job − category that has the largest difference between vacancies (the unfilled − jobs of this category) and the number of jobless people who are − qualified for the jobs of this category.
- 27. Find the courses that can help most jobless people find a job by training them toward the jobs of this category that have the most openings due to lack of qualified workers.
- 28. List all the courses, directly or indirectly required, that a person has to take in order to be qualified for a job of the given category, according to his/her skills possessed and courses taken. (required for graduate students only)4. Find all the jobs a person is currently holding and worked in the past.