-- #1

COLUMN Employee FORMAT A30

COLUMN Times FORMAT A5

COLUMN Earliest\_Date FORMAT A15

SELECT e.emp\_num ||': '|| e.fname ||' '|| e.lname AS "Employee", s.name AS "Skill",

COUNT(s.code) AS "Times", MIN(t.date\_acquired) AS "Earliest\_Date",

TRUNC(MONTHS\_BETWEEN(SYSDATE, MAX(t.date\_acquired))) AS "#Passed\_Month"

FROM employee e, training t, skill s

WHERE e.emp\_num = t.emp\_num

AND t.code = s.code

GROUP BY e.emp\_num, e.fname, e.lname, s.name

ORDER BY e.emp\_num;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #2

COLUMN Employee FORMAT A40

SELECT LEVEL,

LPAD(' ', 3\*(LEVEL-1)) || emp\_num || ' ' || fname || ' ' || lname AS "Employee",

department.name AS "Dept\_Name"

FROM employee JOIN department USING(dept\_code)

START WITH emp\_num=1001

CONNECT BY PRIOR emp\_num = super\_id;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #3

COLUMN start\_date FORMAT A10

COLUMN Proj\_Start FORMAT A10

COLUMN Month FORMAT A6

SELECT a.proj\_number || ': ' || p.name AS "Project", p.start\_date AS "Proj\_Start", TO\_CHAR(a.date\_assigned,'MM') AS "Month",

COUNT(DISTINCT a.emp\_num) AS "#Employees", SUM(a.hours\_used)

FROM project p, assignment a

WHERE p.proj\_number = a.proj\_number

AND p.total\_cost IS NULL

GROUP BY GROUPING SETS((a.proj\_number, p.name, p.start\_date), (a.proj\_number, p.name, p.start\_date, a.date\_assigned));

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #4

ALTER TABLE employee

ADD Bonus\_Amt NUMBER(4,0);

UPDATE employee

SET Bonus\_Amt = CASE WHEN emp\_num IN

(SELECT emp\_num

FROM project JOIN assignment USING(proj\_number)

WHERE EXTRACT(YEAR FROM start\_date) = EXTRACT(YEAR FROM SYSDATE)

AND ROUND(EXTRACT(MONTH FROM start\_date)/3) <= 1

GROUP BY EMP\_NUM

HAVING SUM(hours\_used) >= 150) THEN 200

ELSE 0

END;

SELECT \* FROM employee;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #5

COLUMN date\_acquired FORMAT A15

BREAK ON Employee ON hire\_date

SELECT e.emp\_num ||': '|| e.fname ||' '|| e.lname AS "Employee", e.hire\_date,

t.name AS "Training\_Name", t.date\_acquired, t.date\_acquired - e.hire\_date AS "Days of Training",

COUNT(DISTINCT a.proj\_number) AS "#Porject"

FROM employee e

LEFT JOIN training t

ON e.emp\_num = t.emp\_num

LEFT JOIN assignment a

ON e.emp\_num = a.emp\_num

WHERE e.hire\_date >= TO\_DATE('2021-04-01','yyyy-mm-dd')

AND e.hire\_date <= TO\_DATE('2021-06-30','yyyy-mm-dd')

GROUP BY e.emp\_num, e.fname, e.lname, e.hire\_date, t.name, t.date\_acquired

ORDER BY e.emp\_num;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #6

SELECT name, start\_date,

CASE WHEN total\_cost IS NULL THEN 'on-going'

ELSE 'completed'

END AS "Status"

FROM (SELECT a1.proj\_number, a1.date\_assigned, a2.date\_assigned, p.name, p.start\_date, p.total\_cost

FROM assignment a1

LEFT JOIN assignment a2 on a1.proj\_number = a2.proj\_number

JOIN project p on p.proj\_number = a1.proj\_number

WHERE a1.date\_assigned != a2.date\_assigned

AND a1.date\_assigned > a2.date\_assigned

GROUP BY a1.proj\_number, a1.date\_assigned, a2.date\_assigned, p.name, p.start\_date, p.total\_cost

HAVING MONTHS\_BETWEEN(a1.date\_assigned, a2.date\_assigned) > 1

ORDER BY proj\_number, a1.date\_assigned, a2.date\_assigned);

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #7

COLUMN Quarter FORMAT A10

SELECT TO\_CHAR(p.start\_date,'Q') quarter, COUNT(DISTINCT p.proj\_number) AS "#Project",

COUNT(DISTINCT a.emp\_num) AS "#Employee", SUM(a.hours\_used) / COUNT(DISTINCT p.proj\_number) AVG\_Hours

FROM project p, assignment a

WHERE p.proj\_number = a.proj\_number

AND TO\_CHAR(p.start\_date,'YYYY') = '2021'

GROUP BY TO\_CHAR(p.start\_date,'YYYY') ,TO\_CHAR(p.start\_date,'Q')

ORDER BY TO\_CHAR(p.start\_date,'YYYY') ,TO\_CHAR(p.start\_date,'Q');

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #8

SELECT DECODE(emp\_num, null, '---', emp\_num) ID,

DECODE(emp\_name, null, 'Number of Trainings:', emp\_name) "Employee Name",

NVL(SUM(DECODE(code,01,1)),0) AS Java,

(CASE

WHEN emp\_name IS NULL THEN '------'

ELSE MAX(java\_da)

END

) AS "Latest Date Acquired",

NVL(SUM(DECODE(code,02,1)),0) AS Advertising,

(CASE

WHEN emp\_name IS NULL THEN '------'

ELSE MAX(ad\_da)

END

) AS "Latest Date Acquired",

NVL(SUM(DECODE(code,03,1)),0) AS Writing,

(CASE

WHEN emp\_name IS NULL THEN '------'

ELSE MAX(writing\_da)

END

) AS "Latest Date Acquired",

(CASE

WHEN emp\_name IS NULL THEN '------'

ELSE TO\_CHAR(COUNT(\*))

END

) AS "Number of Skills"

FROM (SELECT emp\_num, fname || ' ' || lname AS emp\_name,

skill.name,

code,

training.date\_acquired,

(CASE

WHEN code = 01 THEN TO\_CHAR(MAX(date\_acquired), 'DD-MON-YY')

ELSE '------'

END

) AS java\_da,

(CASE

WHEN code = 02 THEN TO\_CHAR(MAX(date\_acquired), 'DD-MON-YY')

ELSE '------'

END

) AS ad\_da,

(CASE

WHEN code = 03 THEN TO\_CHAR(MAX(date\_acquired), 'DD-MON-YY')

ELSE '------'

END

) AS writing\_da

FROM employee JOIN training USING(emp\_num)

JOIN skill USING(code)

GROUP BY emp\_num, fname || ' ' || lname, skill.name, training.date\_acquired, code)

GROUP BY GROUPING SETS( (emp\_num, emp\_name), ());

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #9

SELECT d.name AS "Department", s.name AS "Skill", NVL(count, 0) AS "#training",

RANK() OVER (PARTITION BY d.name ORDER BY count DESC NULLS LAST) rank

FROM department d

CROSS JOIN skill s

LEFT JOIN

(SELECT d.name dep\_name, s.name skill\_name, COUNT(t.train\_num) count

FROM skill s

JOIN training t

ON s.code = t.code

JOIN employee e

ON t.emp\_num = e.emp\_num

JOIN department d

ON e.dept\_code = d.dept\_code

GROUP BY d.name, s.name)

ON s.name = skill\_name

AND d.name = dep\_name;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #10

/\*

Nested

- Group the records and sum up the now "top 3 records" and eliminate any groups who have less than 60 days in first 3 records/assignments

- Reduce projects v down to top 3 records

- Get Projects with > 5 assignment records and sum up total days worked on those projects

\*/

SELECT proj\_number, name, Tot\_Days\_Worked

FROM(SELECT \*

FROM project JOIN (SELECT proj\_number, Tot\_Days\_Worked,

(date\_ended-date\_assigned) AS Assign\_Days\_Worked,

ROW\_NUMBER() OVER (PARTITION BY proj\_number ORDER BY date\_assigned) rank

FROM assignment JOIN (SELECT proj\_number, SUM(date\_ended - date\_assigned) AS Tot\_Days\_Worked

FROM assignment

GROUP BY proj\_number

HAVING (COUNT(\*) > 5)

ORDER BY proj\_number)

USING(proj\_number))

USING(proj\_number)

WHERE rank <= 3)

GROUP BY proj\_number, name, tot\_days\_worked

HAVING SUM(Assign\_Days\_Worked) >= 60;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #11

COLUMN lname FORMAT A15

SELECT e.lname AS "Employee", e.hire\_date, d.name AS "Managed\_Dept",

s.supervising AS "#supervising"

FROM employee e

LEFT JOIN department d

ON e.dept\_code = d.dept\_code

LEFT JOIN

(SELECT super\_id, count(\*) supervising

FROM employee

WHERE super\_id is NOT NULL

GROUP BY super\_id) s

ON e.emp\_num = s.super\_id

WHERE rownum <= 4

ORDER BY e.hire\_date;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #12

/\*

1. Map each client to type from web address

2. How many clients in each category

3. how many projects for each category

\*/

SELECT type, NVL(client\_count,0) "Project Count", NVL(proj\_count,0) "Client Count"

FROM (SELECT 'Educational' AS Type FROM dual UNION ALL

SELECT 'Government' AS Type FROM dual UNION ALL

SELECT 'Non-Profit' AS Type FROM dual UNION ALL

SELECT 'For-Profit' AS Type FROM dual UNION ALL

SELECT 'Other' AS Type FROM dual

) type\_table

LEFT JOIN (SELECT type, COUNT(DISTINCT client\_id) client\_count, COUNT(DISTINCT proj\_number) proj\_count

FROM project FULL OUTER JOIN (SELECT (CASE

WHEN LOWER(web\_address) LIKE '%.edu%' THEN 'Educational'

WHEN LOWER(web\_address) LIKE '%.gov%' THEN 'Government'

WHEN LOWER(web\_address) LIKE '%.org%' THEN 'Non-Profit'

WHEN LOWER(web\_address) LIKE '%.com%' THEN 'For-Profit'

ELSE 'Other'

END) Type, client\_id

FROM client)

USING(client\_id)

GROUP BY type)

USING(type);

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #13

COLUMN Last\_project FORMAT A15

SELECT employee, department, proj\_number AS "Last\_Project"

FROM

(SELECT e.emp\_num ||': '|| e.fname ||' '|| e.lname employee, d.name department,

p.proj\_number, p.start\_date,

ROW\_NUMBER() OVER (PARTITION BY e.emp\_num ORDER BY p.start\_date DESC) rn

FROM employee e

LEFT JOIN assignment a

ON e.emp\_num = a.emp\_num

LEFT JOIN project p

ON p.proj\_number = a.proj\_number

JOIN department d

ON e.dept\_code = d.dept\_code

WHERE e.emp\_num NOT IN

(SELECT a.emp\_num

FROM project p, assignment a

WHERE p.proj\_number = a.proj\_number

AND p.start\_date >= TO\_DATE('2021-08-01','yyyy-mm-dd'))

GROUP BY e.emp\_num, e.fname, e.lname, d.name, p.proj\_number, p.start\_date

ORDER BY d.name, e.lname)

WHERE rn = 1;

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #14

--For each skill, list number of employee trainings and number of projects requiring that skill code

SELECT DECODE(skill\_name, null, 'Totals:', skill\_name) "Skill Name",

DECODE(sum\_proj, null, SUM(sum\_proj), sum\_proj) "# of Projects requiring skill",

DECODE(sum\_emp, null, SUM(sum\_emp), sum\_emp) "# of Employee trainings"

FROM( SELECT skill\_name, sum\_proj, COUNT(\*) AS sum\_emp

FROM training LEFT JOIN (select skill.name skill\_name, COUNT(\*) AS sum\_proj, code

FROM project left join skill using(code)

GROUP BY skill.name, code)

USING(code)

group by skill\_name, sum\_proj)

GROUP BY GROUPING SETS((skill\_name, sum\_proj, sum\_emp), ());

--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>--<>

-- #15

SET LINESIZE 300

COLUMN table\_name FORMAT A15

COLUMN column\_name FORMAT A15

COLUMN constraint\_name FORMAT A30

COLUMN search\_condition FORMAT A40

COLUMN FK\_references FORMAT A30

BREAK ON table\_name

SELECT c.table\_name, cc.column\_name, c.constraint\_name,

Case

WHEN c.constraint\_type = 'P' THEN 'PK'

WHEN c.constraint\_type = 'R' THEN 'FK'

WHEN c.constraint\_type = 'C' AND c.constraint\_name LIKE '%\_CK' THEN 'CK'

WHEN c.constraint\_type = 'C' AND c.constraint\_name LIKE '%\_NN' THEN 'NN'

WHEN c.constraint\_type = 'U' THEN 'UK'

ELSE c.constraint\_type

END,

c.search\_condition,

Case c.constraint\_type

WHEN 'R' THEN SUBSTR(c.r\_constraint\_name, 1, INSTR(c.r\_constraint\_name, '\_') - 1)

||':'|| RTRIM(SUBSTR(c.r\_constraint\_name, INSTR(c.r\_constraint\_name, '\_') + 1, LENGTH(c.r\_constraint\_name)), '\_PK')

END AS "FK\_references"

FROM user\_constraints c

JOIN user\_cons\_columns cc

ON c.table\_name = cc.table\_name

AND c.constraint\_name = cc.constraint\_name

ORDER BY c.table\_name, cc.position;