SQL Queries:

[1] Average number of benefits per job and details of jobs offering higher than the average # of benefits

```
## Getting number of benefits in the first CTE (Common Table Expression) ##

with num_benefits as (
        select job_id, count(distinct type) as num_benefits
        from benefits
        group by 1
),
Getting avg number of benefits in the next CTE

avg_benefits as (
select avg(num_benefits) as avg_benefits_count
from num_benefits
)
select n.job_id, jp.company_id, c.name as company_name, jp.title, num_benefits
from num_benefits n
inner join JobPosting jp
on n.job_id = jp.job_id
inner join Companies c
on jp.company_id = c.company_id
where num_benefits >= (select avg_benefits_count from avg_benefits)
order by num_benefits desc;
```

[2] Find the average salary by each job industry. Rank to find the top 3 industries by avg salary (joins and window function)

```
## Getting avg of min_salary for all non-null values by industry ##

with avg_salary_by_industry as (
select industry,  round(avg(min_salary),3) as avg_salary
from JobPosting jp
inner join CompanyIndustries ci
on jp.company_id = ci.company_id
where industry is not null
and coalesce(min_salary,0) > 0
group by 1
),
ranking_salary as (
select industry, avg_salary,
```

```
dense_rank() over (order by avg_salary desc) as salary_rank
from avg_salary_by_industry
)
select * from ranking_salary where salary_rank<=3
order by salary_rank;
```

[3] Find the top skill for each job industry. (Top skills are assumed to be based on # job postings that require those skills)

```
with industry_skill_count as (
        select ci.industry as company_industry, jp.job_id, js.skill_abr
        from JobSkills js
        inner join JobPostings jp
        on js.job_id = jp.job_id
        inner join CompanyIndustries ci
        on jp.company_id = ci.company_id
        group by 1,2,3
),
skill_count_raw as (
select company_industry, skill_abr,
count(distinct job_id) as job_count
from industry_skill_count
group by 1,2
),
skill_ranking as (
select company_industry, skill_abr, job_count,
dense_rank() over (partition by company_industry order by job_count desc) as skill_rank
from skill_count_raw
)
select * from skill_ranking where skill_rank=1
order by company_industry, job_count desc;
```

[4] Which company has shown max employee count growth % over time?

```
with max_min_emp_count_raw as (
select company_id,
## Casting varchar to timestamp in case data type is not already datetime ##
cast(time_recorded as datetime) as date_time_stamp,
employee_count
from employee_counts
where time_recorded is not null
and coalesce(employee_count,0)>0
group by 1,2,3
```

```
),
min_max_rank as (
select company_id, date_time_stamp, employee_count,
## Using row number since we want unique row for a rank, dense rank may show duplicates ##
## Creating 2 ranks - 1 for earliest and 1 for latest ##
row_number() over (partition by company_id order by date_time_stamp) as min_rank,
row_number() over (partition by company_id order by date_time_stamp desc) as max_rank
from  max_min_emp_count_raw
),
growth_calc as (
select coalesce(mn.company_id, mx.company_id) as company_id,
# Including condition to make growth % as 0 if denominator is 0, else do the actual percentage change
case when coalesce(mn.earliest_emp_count,0)>0 then
        (coalesce(mx.latest_emp_count,0) -
coalesce(mn.earliest_emp_count,0))*100/coalesce(mn.earliest_emp_count,0)
  else 0 end as emp_count_growth_change_percentage from
(select company_id, employee_count as earliest_emp_count from min_max_rank where min_rank=1) mn
left join
(select company_id, employee_count as latest_emp_count from min_max_rank where max_rank=1) mx
on mn.company_id = mx.company_id
)
select c.name as company_name, gc.* from
growth_calc gc inner join Companies c
on gc.company_id = c.company_id
order by emp_count_growth_change_percentage desc;
```

[5] Which job industry has the highest number of openings? (Highest number of openings would mean the job_id doesn't have a closed time yet)

```
select * from
(
        select ci.industry as company_industry,
  count(distinct job_id) as jobs_count
        from JobPosting jp
        inner join CompanyIndustries ci
        on jp.company_id = ci.company_id
  where jp.closed_time_ts is not null
        group by 1
  order by jobs_count desc
) s
limit 1;
```

[6] Which companies have the highest number of specialties by industry?

```
with spec_count as (
        select industry, c.company_id, c.name as company_name,
        count(distinct speciality) as specialities_count
        from CompanySpecialities cs inner join Companies c
        on cs.company_id = c.company_id
   inner join CompanyIndustries ci
   on ci.company_id = c.company_id
        group by 1,2,3
),
spec_rank as (
        select sc.*,
        dense_rank() over (partition by industry order by specialities_count desc) as sp_rank
        from spec_count sc
)
select * from spec_rank
where sp_rank=1
order by specialities_count desc;
```

[7] What is the count of internships available with company name, size and job description?

```
select c.name as company_name, c.company_size, jp.description as job_description,
count(distinct job_id) as jobs_count
from JobPosting jp
inner join Companies c
on jp.company_id = c.company_id
where jp.closed_time is not null
and lower(trim(jp.work_type)) = 'internship'
group by 1,2,3
order by jobs_count desc;
```

[8] How many full-time jobs offer medical, dental, and 401k as Benefits and have HQ in California?

```
# Filtering for all jobs with the requried benefits
with required_benefits_jobs as (
        select job_id
        from Benefits
        where lower(trim(type)) in ('401k', 'medical insurance', 'dental insurance')
        group by 1
)
select count(distinct jp.job_id) as jobs_count
from JobPosting jp inner join Companies c
```

```
on jp.company_id = c.company_id
inner join required_benefits_jobs ben
on jp.job_id = ben.job_id
where lower(trim(c.state)) in ('ca', 'california');
```

[9] Calculate average, maximum and minimum salaries for companies that are part of 'Nonprofit Organization Management' industry by each state

```
WITH ITCompanies AS (
   SELECT c.state, jp.max_salary, jp.min_salary
   FROM Companies c
   JOIN CompanyIndustries ci ON c.company_id = ci.company_id
   JOIN JobPosting jp ON c.company_id = jp.company_id
   WHERE ci.industry = 'Nonprofit Organization Management'
)

SELECT state,
      max_salary AS max_salary,
      min_salary AS min_salary,
      AVG(max_salary) OVER (PARTITION BY state) AS avg_max_salary,
      AVG(min_salary) OVER (PARTITION BY state) AS avg_min_salary
FROM ITCompanies;
```

[10] List all companies in a specific city (ex: New York) and their average employee counts.

```
WITH CompanyEmployeeCounts AS (
   SELECT c.name, c.city, ec.employee_count,
         AVG(ec.employee_count) OVER (PARTITION BY c.city) AS avg_employee_count
   FROM Companies c
   LEFT JOIN EmployeesCount ec ON c.company_id = ec.company_id
)
SELECT name, city, avg_employee_count
FROM CompanyEmployeeCounts
WHERE city = 'New York';
```

[11] Find companies with a specific specialty (ex: Financial Services) that also have job postings with a certain skill (ex: ACCT) requirement:

```
WITH CompanyJobSkills AS (
   SELECT c.name as company_name, cs.speciality, js.skill_abr,
         ROW_NUMBER() OVER (PARTITION BY c.name, cs.speciality, js.skill_abr) AS rn
   FROM Companies c
   JOIN CompanySpecialities cs ON c.company_id = cs.company_id
   JOIN JobPosting jp ON c.company_id = jp.company_id
```

```
   JOIN JobSkills js ON jp.job_id = js.job_id
)
SELECT 'Job Roll', speciality, skill_abr
FROM CompanyJobSkills
WHERE speciality = 'Financial Services' AND skill_abr = 'ACCT'
 AND rn = 1
ORDER BY company_name;
```

STORED PROCEDURE

[1] GetSkillsAbrByCompany

```
DELIMITER //
CREATE PROCEDURE GetSkillsAbrByCompany(IN companyID_input BIGINT)
BEGIN
  SELECT name as company_name, skill_abr, COUNT(js.job_id) AS job_count
  FROM JobSkills js JOIN JobPosting jp ON js.job_id =jp.job_id
  JOIN Companies c ON jp.company_id = c.company_id
  WHERE c.company_id = companyID_input
  GROUP BY skill_abr, company_name
  ORDER BY job_count DESC;
END //
DELIMITER ;
```

Example of usage: call GetSkillsAbrByCompany(1016);

[2] UpdateJobPostingSalary (Updating new salary to max salary)

```
DELIMITER $$

CREATE PROCEDURE UpdateJobPostingSalary1(IN p_job_ID BIGINT, newSalary DECIMAL(10,2))
BEGIN
        UPDATE JobPosting
   SET max_salary = newSalary
   WHERE job_id = p_job_ID;
END $$

DELIMITER ;

SET SQL_SAFE_UPDATES = 0;
```

Example of usage: call UpdateJobPostingSalary1(133114754, 80000.00);

[3] GetJobPostingsByLocation

```
DELIMITER //
CREATE PROCEDURE GetJobPostingsByLocation(IN joblocation VARCHAR(200))
BEGIN
   SET @joblocation = joblocation;
   SELECT job_id, title FROM JobPosting WHERE location = @joblocation;
END //
DELIMITER ;
```

Example of usage: CALL GetJobPostingsByLocation('New York NY');

TRIGGERS

[1] BeforeJobPostingUpdate

```
DELIMITER //
CREATE TRIGGER BeforeJobPostingUpdate
BEFORE UPDATE ON Jobposting
FOR EACH ROW
BEGIN
  IF NEW.expiry < OLD.expiry THEN
    SET NEW.closed_time = NOW();
  END IF;
END;
//
DELIMITER ;
```

[2] AfterJobPostingInsert

```
DELIMITER //
CREATE TRIGGER AfterJobPostingInsert
AFTER INSERT ON Jobposting
FOR EACH ROW
BEGIN
  INSERT INTO JobPostingLog (job_id, company_id, title, inserted_at)
  VALUES (NEW.job_id, NEW.company_id, NEW.title, NOW());
END;
//
DELIMITER ;
```

[3] AfterSkillsUpdate

```
DELIMITER //
CREATE TRIGGER AfterSkillsUpdate
AFTER UPDATE ON Jobskills
FOR EACH ROW
BEGIN
  INSERT INTO NotificationLog (message, recipient, sent_at)
  VALUES ('Skills for job ' or NEW.job_id or ' have been updated.', 'HR Department', NOW());
END;
//
DELIMITER ;
```

[4] AfterSkillsForRemoteWorkingInsert

```
DELIMITER //
CREATE TRIGGER AfterSkillsForRemoteWorkingInsert
AFTER INSERT ON Jobskills
FOR EACH ROW
BEGIN
  IF NEW.skill_abr = 'RemoteWork' THEN
    INSERT INTO NotificationLog (message, recipient, sent_at)
    VALUES ('Remote work skills added for job ' OR NEW.job_id, 'Remote Work Department', NOW());
  END IF;
END;
//
DELIMITER ;
```