SQL Basics

SELECT columns
FROM table
WHERE conditions
GROUP BY columns
ORDER BY column
HAVING groupy_condition
LIMIT value

- **AS is used to assign alias to column name or table name .**
- **DISTINCT as a standalone keyword after SELECT keyword to remove duplicate rows.**
- **SELECT clause can also have aggregate functions like COUNT(), MAX(), MIN(), SUM(), AVG() which are usually used with GROUP BY clause.**
- **GROUP BY clause contains name of categorical (non-numeric) column(s) like gender, department, city based on which we can create subsets, and on each of these subsets we wish to apply some aggregate function.**
- **ORDER BY clause contains name of column on which the table is to be sorted, we can use ASC or DESC keyword after column name (if we don't specify ASC or DESC then by default ASC is taken).**
- **HAVING is used to apply a condition on aggregate function.**
- **LIMIT is used to specify the number of rows to be displayed.**
- **WHERE is used to specify conditions which can include operators =, >, <, +, -, etc. and keywords like AND, OR, BETWEEN, IN, LIKE, etc.**
- **IN is used to search a value in a list e.g. city IN ('Delhi', 'Mumbai')**
- **LIKE is used for string matching with regular expressions e.g. LIKE 'A%'.**
- **FROM clause can also join multiple tables using INNER JOIN, etc. on a common column using ON keyword.**
- **UNION can be used to merge rows of two tables.**
- **MINUS can be used to remove a subset of rows from a superset of rows.**
- **YEAR(date), MONTH(date), DAY(date), DATE(timestamp) can extract desired part from a timestamp.**
- **DATEDIFF() will return number of days between two dates.**
- **CURDATE() function can be used to obtain current date.**
- **We can extract a range of dates starting from a fixed date using INTERVAL keyword e.g. CURDATE() – INTERVAL 2 DAY**
- **LAG(c1,2) AS lag_c1 will create a new column lag_c1 having same values as column c1 but with a shift of 2, the first two values will be blank by default which can be set using third parameter of LAG.**
- **PARTITION BY is used with aggregate/lag functions, however it displays all rows of each partition e.g. AVG() OVER(PARTITION BY column).**
- **DENSE_RANK() OVER (ORDER BY profit DESC) AS position will rank all the elements.**
- **CASE WHEN condition THEN action WHEN condition THEN action ELSE action END AS column_name**

Report the movies with an odd-numbered ID and whose description is not boring.
SELECT * FROM movies WHERE movie_id%2>0 AND description != 'boring'

Count the number of movies that Abigail Breslin was nominated for an oscar.
SELECT COUNT(*) AS n_movies_by_abi
FROM oscar_nominees WHERE nominee = 'Abigail Breslin'

Find duplicate mails.
SELECT emails FROM persons GROUP BY emails HAVING COUNT(*)>1

Find second highest salary.
SELECT name, MAX(salary) AS salary FROM employee WHERE salary IN
(SELECT salary FROM employee MINUS SELECT MAX(salary) FROM employee)

Find highest salary per department.
SELECT department, MAX(salary) AS salary
FROM employee GROUP BY department ORDER BY salary DESC
Note: each column in select clause must either be inside an aggregate function or also in group by clause.

Find the employee with the highest salary per department. Output the department name, employee's first name along with the corresponding salary.
SELECT department, first_name, salary FROM employee
WHERE (department , salary) IN (SELECT department, MAX(salary) FROM employee GROUP BY department)

Find the review_text that received the highest number of 'cool' votes. Output the business name along with the review text with the highest numbef of 'cool' votes.
yelp_reviews(business_name:varchar, review_id:varchar, user_id:varchar, stars:varchar, review_date:datetime, review_text:varchar, funny:int, useful:int, cool:int)
SELECT business_name, review_text
FROM yelp_reviews
WHERE cool = (SELECT MAX(cool) FROM yelp_reviews)

Find employees who are earning more than their managers. Output the employee's first name along with the corresponding salary.
employee(id:int, first_name:varchar, last_name:varchar, age:int, sex:varchar, employee_title:varchar, department:varchar, salary:int, target:int, bonus:int, email:varchar, city:varchar, address:varchar, manager_id:int)
SELECT a.first_name AS employee_name, a.salary AS employee_salary
FROM employee AS a JOIN employee AS b ON a.manager_id = b.id
WHERE a.salary > b.salary;

Find matching hosts and guests pairs in a way that they are both of the same gender and nationality. Output the host id and the guest id of matched pair.
airbnb_hosts(host_id:int, nationality:varchar, gender:varchar, age:int)
SELECT DISTINCT h.host_id, g.guest_id
FROM airbnb_hosts h INNER JOIN airbnb_guests g ON h.nationality = g.nationality AND h.gender = g.gender

Easy

E1) Find the activity date and the pe_description of facilities with the name 'STREET CHURROS' and with a score of less than 95 points.
los_angeles_restaurant_health_inspections(serial_number:varchar, activity_date:datetime, facility_name:varchar, score:int, grade:varchar, service_code:int, service_description:varchar, employee_id:varchar, facility_address:varchar, facility_city:varchar, facility_id:varchar, facility_state:varchar, facility_zip:varchar, owner_id:varchar, owner_name:varchar, pe_description:varchar, program_element_pe:int, program_name:varchar, program_status:varchar, record_id:varchar)

E2) Find libraries who haven't provided the email address in circulation year 2016 but their notice preference definition is set to email. Output the library code.
library_usage(patron_type_code:int, patron_type_definition:varchar, total_checkouts:int, total_renewals:int, age_range:varchar, home_library_code:varchar, home_library_definition:varchar, circulation_active_month:varchar, circulation_active_year:float, notice_preference_code:varchar, notice_preference_definition:varchar, provided_email_address:bool, year_patron_registered:int, outside_of_county:bool, supervisor_district:float)

E3) Find the average number of bathrooms and bedrooms for each city's property types. Output the result along with the city name and the property type.
airbnb_search_details(id:int, price:float, property_type:var, charroom_type:varchar, amenities:varchar, accommodates:int, bathrooms:int, bed_type:varchar, cancellation_policy:varchar, cleaning_fee:bool, city:varchar, host_identity_verified:varchar, host_response_rate:varchar, host_since:datetime, neighbourhood:varchar, number_of_reviews:int, review_scores_rating:float, zipcode:int, bedrooms:int, beds:int)

E4) Find the last time each bike was in use. Output both the bike number and the date-timestamp of the bike's last use (i.e., the date-time the bike was returned). Order the results by bikes that were most recently used.
dc_bikeshare_q1_2012(duration:varchar, duration_seconds:int, start_time:datetime, start_station:varchar, start_terminal:int, end_time:datetime, end_station:varchar, end_terminal:int, bike_number:varchar, rider_type:varcharid:int)

E5) Find how many times each artist appeared on the Spotify ranking list Output the artist name along with the corresponding number of occurrences. Order records by the number of occurrences in descending order.
spotify_worldwide_daily_song_ranking(id:int, position:int, trackname:varchar, artist:varchar, streams:int, url:varchar, date:datetime, region:varchar)

E6) We have a table with employees and their salaries, however, some of the records are old and contain outdated salary information. Find the current salary of each employee assuming that salaries increase each year. Output their id, first name, last name, department ID, and current salary. Order your list by employee ID in ascending order.
ms_employee_salary(id:int, first_name:varchar, last_name: varchar, salary:int, department_id:int)

E7) Count the number of user events performed by MacBookPro users. Output the result along with the event name. Sort the result based on the event count in the descending order.
playbook_events(user_id:int, occurred_at:datetime, event_type:varchar, event_name:varchar, location:varchar, device:varchar)

E8) Find order details made by Jill and Eva. Consider the Jill and Eva as first names of customers. Output the order date, details and cost along with the first name. Order records based on the customer id in ascending order.
customers(id:int, first_name:varchar, last_name:varchar, city:varchar, address:varchar, phone_number:varchar)
orders(id:int, cust_id:int, order_date:datetime, order_details:varchar, total_order_cost:int)

E9) Find all posts which were reacted to with a heart. Heart reactions have the value of 'heart' for the reaction column of the facebook_reactions dataset.
facebook_reactions(poster:int, friend:int, reaction:varchar, date_day:int, post_id:int)
facebook_posts(post_id:int, poster:int, post_text:varchar, post_keywords:varchar, post_date:datetime)

E10) Meta/Facebook has developed a new programing language called Hack.To measure the popularity of Hack they ran a survey with their employees. The survey included data on previous programing familiarity as well as the number of years of experience, age, gender and most importantly satisfaction with Hack. Due to an error location data was not collected, but your supervisor demands a report showing average popularity of Hack by office location. Luckily the user IDs of employees completing the surveys were stored. Based on the above, find the average popularity of the Hack per office location. Output the location along with the average popularity.
facebook_employees(id:int, location:varchar, age:int, gender:varchar, is_senior:bool)
facebook_hack_survey(employee_id:int, age:int, gender:varchar, popularity:int)

E11) Find the details of each customer regardless of whether the customer made an order. Output the customer's first name, last name, and the city along with the order details. You may have duplicate rows in your results due to a customer ordering several of the same items. Sort records based on the customer's first name and the order details in ascending order.
customers(id:int, first_name:varchar, last_name:varchar, city:varchar, address:varchar, phone_number:varchar)
orders(id:int, cust_id:int, order_date:datetime, order_details:varchar, total_order_cost:int)

E12) Find the titles of workers that earn the highest salary. Output the highest-paid title or multiple titles that share the highest salary.
worker(worker_id:intfirst_name:varcharlast_name:varcharsalary:intjoining_date:datetimedepartment:varchar)
title(worker_ref_id:intworker_title:varcharaffected_from:datetime)

E13) Find the most profitable company from the financial sector. Output the result along with the continent.
forbes_global_2010_2014(company:varchar, sector:varchar, industry:varchar, continent:varchar, country:varchar, marketvalue:float, sales:float, profits:float, assets:float, rank:int, forbeswebpage:varchar)

E14) Write a query that calculates the difference between the highest salaries found in the marketing and engineering departments. Output just the absolute difference in salaries.
db_employee(id:int, first_name:varchar, last_name:varchar, salary:int, department_id:int, email:datetime)
db_dept(id:int, department:varchar)

ANSWER-E1

```
SELECT activity_date, pe_description
FROM los_angeles_restaurant_health_inspections
WHERE facility_name='STREET CHURROS' AND score<95;
```

ANSWER-E2

```
SELECT DISTINCT home_library_code
FROM library_usage
WHERE notice_preference_definition = 'email' AND provided_email_address is FALSE AND circulation_active_year = 2016
```

ANSWER-E3

```
SELECT city, property_type, AVG(bathrooms) AS n_bathrooms_avg, AVG(bedrooms) AS n_bedrooms_avg
FROM airbnb_search_details
GROUP BY city, property_type
```

ANSWER-E4

```
SELECT bike_number, MAX(end_time) AS last_used
FROM dc_bikeshare_q1_2012
GROUP BY bike_number
ORDER BY last_used DESC
```

ANSWER-E5

```
SELECT artist, COUNT(artist) AS artist_count
FROM spotify_worldwide_daily_song_ranking
GROUP BY artist
ORDER BY artist_count DESC
```

ANSWER-E6

```
SELECT id, first_name, last_name, department_id, MAX(salary)
FROM ms_employee_salary
GROUP BY id, first_name, last_name, department_id
ORDER BY id ASC
```

ANSWER-E7

```
SELECT event_name, COUNT(*) AS event_count
FROM playbook_events
WHERE device = 'macbook pro'
GROUP BY event_name
ORDER BY event_count DESC
```

ANSWER-E8

```
SELECT customers.first_name, order_date, order_details, total_order_cost
FROM orders JOIN customers ON customers.id = orders.cust_id
WHERE customers.first_name IN ('Jill', 'Eva') ORDER BY cust_id
```

ANSWER-E9

```
SELECT DISTINCT fp.*
FROM facebook_reactions AS fr INNER JOIN facebook_posts AS fp ON fr.post_id=fp.post_id
WHERE fr.reaction='heart'
```

ANSWER-E10

```
SELECT e.location, AVG(s.popularity) AS avg_popularity
FROM facebook_employees AS e JOIN facebook_hack_survey AS s ON e.id = s.employee_id
GROUP BY e.location
```

ANSWER-E11

```
SELECT c.first_name, c.last_name, c.city, o.order_details
FROM customers AS c LEFT JOIN orders AS o ON o.cust_id = c.id
ORDER BY c.first_name, o.order_details
```

ANSWER-E12

```
SELECT t.worker_title
FROM worker AS w INNER JOIN title AS t ON w.worker_id=t.worker_ref_id
WHERE salary = (SELECT MAX(salary) FROM worker)
```

ANSWER-E13

```
SELECT company, continent
FROM forbes_global_2010_2014
WHERE sector = 'Financials' AND
profits =  (SELECT MAX(profits) FROM forbes_global_2010_2014 WHERE sector = 'Financials')
```

ANSWER-E14

```
SELECT ABS(
(SELECT MAX(salary) FROM db_employee emp JOIN db_dept dept ON emp.department_id = dept.id
WHERE department = 'marketing') -
(SELECT MAX(salary) FROM db_employee emp JOIN db_dept dept ON emp.department_id = dept.id
 WHERE department = 'engineering')
) AS salary_difference
```