1. **SQL Query to find second highest salary of Employee.**

SELECT name, salary FROM employees e1 WHERE 2-1 = (SELECT COUNT(DISTINCT salary) FROM employees e2 WHERE e2.salary > e1.salary)

**Pros:**

The generic solution works in all database including Oracle, MySQL, SQL SERVER and PostgreSQL.

**Cons:**

Slow, because the inner query will run for every row processed by the outer query.

1. **Difference between UNION and UNION ALL**
2. The main difference between them is that UNION doesn't include duplicate record but UNION ALL does. Since you mostly don't want duplicate rows, UNION is preferred over UNION ALL in reporting and application development. By the way, you should keep in mind that UNION ALL performance better than UNION because it doesn't have to remove duplicate, so no extra work.
3. Another difference between them is that UNION ALL is faster than UNION but may look slow because it returns more data which takes more time to travel via the network.

Select emp\_id, emp\_name from employee;

UNION/UNION ALL

Select cus\_id, cus\_name from Customer;

In order to combine results of two queries, they must contain same number of columns.  
For example if one query contains 3 columns and other contains 4 columns then you can not use UNION or UNION ALL.

1. **Difference between Primary key and Unique key**
2. Unique key in a table can be null, at-least one but primary key can not be null in any table in relation database like MySQL , Oracle etc.
3. There can be only one primary key per table in relation database e.g. MySQL, Oracle or Sybase but there can be more than one unique key per table.
4. Primary Key is used to identify a row (record) in a table whereas Unique-key is to prevent duplicate values in a column.
5. **Difference between Primary and Foreign key**
6. The primary key is unique in the table. So a table cannot contain more than one row with the same primary key, but that's not required for foreign key. You can have more than one rows in a table with same foreign key.
7. Foreign key helps to maintain the integrity of related tables in the database. For example, it's not possible to delete a department, unless a single employee is referring that. So, an accidental delete of such department will be prevented by database management systems.
8. **TRUNCATE vs DELETE in SQL**
9. The first and most important difference between TRUNCATE and DELETE command in SQL is that truncate doesn't log row level details while delete is logged. Since TRUNCATE is not logged, it's not possible to rollback it e.g. in Oracle. But some database may provide rollback functionality for truncate, on the other hand, DELETE is always logged and can be rolled back.
10. Due to above reason, TRUNCATE command is much faster than DELETE, and due to same reason, you should not use DELETE to remove large data set, since every delete operation is logged, it may be possible that your log segment gets filled and blew up, if you try to empty a large table using DELETE command.
11. Truncate is usually used to remove all data from tables e.g. in order to empty a particular table and you can define any filter based upon WHERE clause, but with DELETE, you can define the condition in WHERE clause for removing data from tables.
12. Since TRUNCATE is a DDL operation, it's automatically get committed, on the other hand, DELETE is not auto commit.
13. DELETE command work at row level and acquire the row level lock while TRUNCATE locks the whole table, instead of individual rows.
14. **How to find duplicate records in a table on database**

Problem of duplicates in database arise when you don't have a primary key or unique key on database. Anyway its easy to find duplicate records in table by using group by clause of ANSI SQL. Group by clause is used to group data based upon any column or a number of columns. Once you have grouped data you can filter out duplicates by using having clause. Having clause is counter part of where clause for aggregation queries.

Following SELECT query will only find duplicates records based on name

**select name, count(name) from contacts group by name;**

name  | count(name) |

+-------+-------------+

| Harry |           4 |

| James |           4 |

This is the correct way of finding duplicate contacts at it look both name and phone number

select name, count(name) from contacts group by name, phone;

having clause in SQL query will filter duplicate records from non duplicate records.

select name, count(name) as times from contacts group by name, phone having times>1;