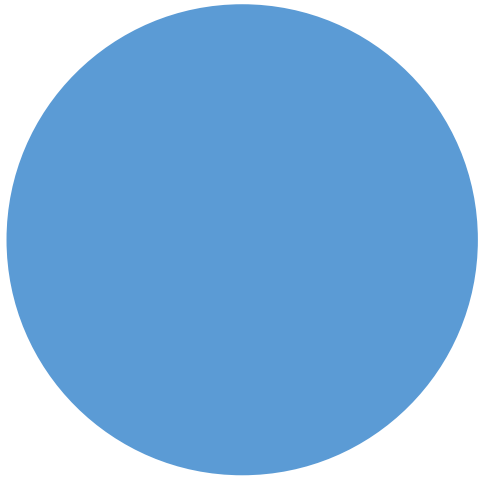
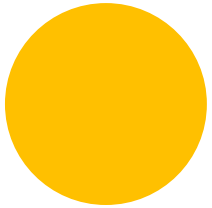
DSC 90

Winter 2020

Interview Questions

SQL and Databases

Types of joins

Explain the difference between a left join and an inner join

Types of joins - discussion

Good question for an early screener for more junior roles since it isn't a trick and is an important knowledge for a candidate to have.

You can learn a lot from how the candidate chooses to answer because there are plenty of valid answers from ones that are textbook correct but not easy to understand to very simple to understand ones that miss edge cases (duplicates, for example)

What are some different ways you can load data into a database in the first place, and what are the advantages and disadvantages of each?

Loading data into SQL

Loading data into SQL - discussion

This question is really a test of whether you've had to load data into a database before or not. If you've done it before then it shouldn't be difficult to describe how you did it. If you haven't loaded data into a database before that might be a signal to an interviewer that you don't have enough experience.

The part of the question about advantages and disadvantages is to check that you understand that different tools are better in different situations. Sometimes using a GUI to upload data is a nice easy solution when you have a single file. Other times you'll want to set up a whole automated script to load the data continuously. The more you can show you understand the nuance of what to use when, the better.

Example SQL query

Here is TABLE_A from a school containing the grades, from 0 to 100, earned by students across multiple classes. How would you calculate the highest grade in each class?

What if we wanted to not only find the highest grade in each class, but also the student who earned that grade?

CLASS	STUDENT	GRADE
Math	Nolis, Amber	100
Math	Berkowitz, Mike	90
Literature	Liston, Amanda	97
Spanish	Betancourt, Laura	93
Literature	Robinson, Abby	93
...	...	...

Nth Highest Salary

Write a SQL query to get the n^{th} highest salary from the table:

ID	Salary
1	100
2	200
3	125
…	…

For example, given the above Employee table, the nth highest salary where n = 2 is 200. If there is no nth highest salary, then the query should return null.

Consider P1(a,b) and P2(c,d) to be two points on a 2D plane.

- a happens to equal the minimum value in Northern Latitude (LAT_N in STATION).
- b happens to equal the minimum value in Western Longitude (LONG_W in STATION).
- c happens to equal the maximum value in Northern Latitude (LAT_N in STATION).
- d happens to equal the maximum value in Western Longitude (LONG_W in STATION).

Query the Manhattan Distance between points P1 and P2 and round it to a scale of 4 decimal places.

Input Format

The STATION table is described as follows:

STATION

Field	Type
ID	NUMBER
CITY	VARCHAR2(21)
STATE	VARCHAR2(2)
LAT_N	NUMBER
LONG_W	NUMBER

where LAT_N is the northern latitude and LONG_W is the western longitude.

Example manipulating data in R/Python

Here's a table, called tweets. The data has the account that sent the tweet, the text, the number of likes, and the date sent. Write a script to get a table with one row per person with a column that's the minimum number of likes they got, called `min_likes`, and a column of the total number of tweets, called `nb_tweets`. This should only be for tweets sent after September 1st, 2019. You also need to eliminate any duplicates in the table first.

account_name	text	nb_likes	date
@vboykis	Data science is...	50	2019-10-01
@Randy_Au	It's hard when...	23	2019-05-01
@rchang	Some news...	35	2019-01-01
@vboykis	My newsletter...	42	2019-11-23
@drob	My best advice...	62	2019-11-01
...	...	...	...

Hard Whiteboarding

Given a string S and a string T, find the minimum window in S which will contain all the characters in T in complexity O(n).

Example:

```
Input: S = "ADOBECODEBANC", T = "ABC"
Output: "BANC"
```

Note:

- If there is no such window in S that covers all characters in T, return the empty string `""`.
- If there is such window, you are guaranteed that there will always be only one unique minimum window in S.

Given an array *nums*, there is a sliding window of size *k* which is moving from the very left of the array to the very right. You can only see the *k* numbers in the window. Each time the sliding window moves right by one position. Return the max sliding window.

Example:

```
Input: nums = [1,3,-1,-3,5,3,6,7], and k = 3
Output: [3,3,5,5,6,7]
Explanation:

Window position                Max
---------------               -----
[1  3  -1] -3  5  3  6  7       3
 1 [3  -1  -3] 5  3  6  7       3
 1  3 [-1  -3  5] 3  6  7       5
 1  3  -1 [-3  5  3] 6  7       5
 1  3  -1  -3 [5  3  6] 7       6
 1  3  -1  -3  5 [3  6  7]      7
```

Note:

You may assume *k* is always valid, 1 ≤ k ≤ input array's size for non-empty array.

Follow up:

Could you solve it in linear time?

Remove the minimum number of invalid parentheses in order to make the input string valid. Return all possible results.

Note: The input string may contain letters other than the parentheses `(` and `)`.

Example 1:

```
Input: "()())()"
Output: ["()()()", "(())()"]
```

Example 2:

```
Input: "(a)())()"
Output: ["(a)()()", "(a())()"]
```

Example 3:

```
Input: ")("
Output: [""]
```

Given two words *word1* and *word2*, find the minimum number of operations required to convert *word1* to *word2*.

You have the following 3 operations permitted on a word:

1. Insert a character
2. Delete a character
3. Replace a character

Example 1:

```
Input: word1 = "horse", word2 = "ros"
Output: 3
Explanation:
horse -> rorse (replace 'h' with 'r')
rorse -> rose (remove 'r')
rose -> ros (remove 'e')
```

Example 2:

```
Input: word1 = "intention", word2 = "execution"
Output: 5
Explanation:
intention -> inention (remove 't')
inention -> enention (replace 'i' with 'e')
enention -> exention (replace 'n' with 'x')
exention -> exection (replace 'n' with 'c')
exection -> execution (insert 'u')
```