Applicant Test: Back-End Web Applications Developer

API Integration

Carefully review the attached API documentation (api/user-api.pdf) and answer the questions below using one of the following languages: JavaScript, PHP, Python, or Ruby. PHP is preferred, but not required.

1. Write a function that calls the production *create user* API method. Return the result of the request as an object. Be sure to properly handle all error cases by returning the appropriate error message.
2. Assuming the user_id is 23234, write a function that calls the development *get user* API method. Return the result of the request as an object. Be sure to properly handle all error cases by returning the appropriate error message.
3. Assuming the user_id is 23234, write a function that calls the production *delete user* API method. Return the result of the request as an object with a 'result' Boolean indicating the outcome of the request and an 'error' string indicating the error, if present.
4. Assuming the user_id is 23234, write a function that calls the production *update user* API method that updates the user's address information and password with an application/json body. Return the result of the request as an object with a 'result' Boolean indicating the outcome of the request and an 'error' string indicating the error, if present.

Hint: be sure to include any necessary credentials that may be required for each API method

Database

Carefully review the attached database schema (database/hockey.png) and SQL file (database/hockey.sql) and answer the following questions.

1. Write a query to return the full name, id, position, total goals, and signed date for all active players on the team 'dallas penguins'.
2. Write a query to return the top 5 days in which the most goals were scored.
3. Write a query to return the full name, id, career length, and team for all retired player. Ordered the results by team name alphabetically from a-z and player name alphabetically from z-a.
4. Write a query to return the full name, position, and total goals scored for all active players on all teams. Order the results by team, position, and then descending by total goals scored.
5. Write a query to determine which position has scored the most overall goals in the year 2015.
6. Write a query to return the top 10 teams who have scored the most goals in the past 5 years.
7. Write a query to return the total goals scored by each retired defensive player on team 'michigan minutemen'.
8. Write a query to return the team that has the most goalie goals overall.

9. Build an index to efficiently return the full name and signed date for all players when searching by players last name.

10. Are there any suggestions you would make to make the schema more efficient?

11. Given the following unique values for each column (you can assume the dates are relatively uniform year-to-year):

first_name	last_name	FK_position	signed_date	total_rows
92345	292343	5	39420	887374

    a. Build an index to most efficiently return results when searching with the where clause:
    b. 'WHERE first_name = 'kevin' AND last_name = 'smith' AND FK_position = 'defense' AND signed_date BETWEEN ('2014-10-01') AND ('2016-12-14');'