to perform database query in WordPress

o perform database queries in WordPress, the primary method involves using the global `$wpdb` object, which is an instance of the `wpdb` class. This class provides a set of methods for interacting with the WordPress database safely and efficiently.

Here's a breakdown of common functions and their usage: Accessing the $wpdb object.

Before using any `$wpdb` methods, you need to declare the global `$wpdb` object within your function or code block.

Code

```
global $wpdb;
```

Preparing Queries for Security.

It is crucial to use `$wpdb->prepare()` to prevent SQL injection vulnerabilities when including dynamic values in your queries.

Code

```
$query = $wpdb->prepare(
    "SELECT * FROM {$wpdb->prefix}posts WHERE post_status = %s AND post_type = %s",
    'publish',
    'post'
);
```

`%s` is a placeholder for strings.

`%d` is a placeholder for integers.

`%f` is a placeholder for floats.

`{$wpdb->prefix}` ensures you are using the correct table prefix for your WordPress installation.

Executing Queries and Retrieving Data:

`$wpdb->get_results()`: Retrieves multiple rows as an array of objects or associative arrays.

Code

```
$results = $wpdb->get_results( $query );
// or
$results = $wpdb->get_results( $query, ARRAY_A ); // for associative arrays
```

`$wpdb->get_row()`: Retrieves a single row as an object or associative array.

Code

```
$row = $wpdb->get_row( $query );
```

`$wpdb->get_var()`: Retrieves a single variable (e.g., a count).

Code

```
$user_count = $wpdb->get_var( "SELECT COUNT(*) FROM {$wpdb->users}" );
```

`$wpdb->get_col()`: Retrieves a single column from multiple rows as an array.

Code

```
$post_titles = $wpdb->get_col( "SELECT post_title FROM {$wpdb->posts}" );
```

`$wpdb->query()`: Executes a generic SQL query (e.g., INSERT, UPDATE, DELETE). This method returns the number of affected rows for manipulation queries or `true`/`false` for other queries.

Code

```
$rows_affected = $wpdb->query( $wpdb->prepare(
    "DELETE FROM {$wpdb->postmeta} WHERE post_id = %d AND meta_key = %s",
    13,
    'stars'
));
```

Inserting, Updating, and Deleting Data (Convenience Methods):

The `$wpdb` class also offers convenience methods for common data manipulation tasks, which internally use `$wpdb->prepare()` for security.

`$wpdb->insert()`: Inserts data into a table.

Code

```
$wpdb->insert(
    "{$wpdb->prefix}mytable",
    array(
        'column1' => 'value1',
        'column2' => 'value2',
    ),
    array( '%s', '%d' ) // Format of values
);
```

`$wpdb->update()`: Updates data in a table.

Code

```
$wpdb->update(
    "{$wpdb->prefix}mytable",
    array( 'column1' => 'new_value' ), // Data to update
    array( 'id' => 1 ), // WHERE clause
    array( '%s' ), // Format of updated values
    array( '%d' ) // Format of WHERE clause values
);
```

`$wpdb->delete()`: Deletes data from a table.

Code

```
$wpdb->delete(
    "{$wpdb->prefix}mytable",
    array( 'id' => 1 ), // WHERE clause
    array( '%d' ) // Format of WHERE clause values
);
```

Revision #2
Created 29 October 2025 02:43:37 by AI API
Updated 1 December 2025 14:03:36 by AI Channel