```python
import threading
import configparser

# Replace with your specific database connector
import psycopg2  # for database connectior for python

class PropertyUtil:
    @staticmethod
    def get_property_string(property_file_name):
        config = configparser.ConfigParser()
        config.read(property_file_name)

        try:
            hostname = config.get("database", "hostname")
            dbname = config.get("database", "dbname")
            username = config.get("database", "username")
            password = config.get("database", "password")
            port = config.getint("database", "port")

            connection_string = f"postgresql://{username}:{password}@{hostname}:{port}/{dbname}"  # Format for psycopg2
            return connection_string

        except (configparser.NoSectionError, configparser.NoOptionError) as e:
            raise PropertyFileException(f"Error reading property file: {e}")

class DBConnection:
    _connection = None
    _lock = threading.Lock()

    @staticmethod
    def get_connection():
        with DBConnection._lock:
            if DBConnection._connection is None:
                try:
                    connection_string = PropertyUtil.get_property_string("database.properties")
                    DBConnection._connection = psycopg2.connect(connection_string)  # Use the actual connector
                except (PropertyFileException, psycopg2.Error) as e:
                    raise DBConnectionException(f"Failed to connect to database: {e}")
            return DBConnection._connection

    @staticmethod
    def close_connection():
        with DBConnection._lock:
            if DBConnection._connection is not None:
                DBConnection._connection.close()
                DBConnection._connection = None

class DBConnectionException(Exception):
    pass

class PropertyFileException(Exception):
    pass
```