```python
# Package: dao

from abc import ABC, abstractmethod
from typing import List
from entity import Appointment
from exception import AppointmentNotFoundException
import sqlalchemy

class IHospitalService(ABC):
    @abstractmethod
    def get_appointment_by_id(self, appointment_id: int) -> Appointment:
        """Retrieves an appointment by its ID."""
        pass

    @abstractmethod
    def get_appointments_for_patient(self, patient_id: int) -> List[Appointment]:
        """Retrieves appointments for a patient."""
        pass

    @abstractmethod
    def get_appointments_for_doctor(self, doctor_id: int) -> List[Appointment]:
        """Retrieves appointments for a doctor."""
        pass

    @abstractmethod
    def schedule_appointment(self, appointment: Appointment) -> bool:
        """Schedules a new appointment."""
        pass

    @abstractmethod
    def update_appointment(self, appointment: Appointment) -> bool:
        """Updates an existing appointment."""
        pass

    @abstractmethod
    def cancel_appointment(self, appointment_id: int) -> bool:
        """Cancels an appointment."""
        pass

class HospitalServiceImpl(IHospitalService):

    def __init__(self, db_util):
        self.db_util = db_util

    def get_appointment_by_id(self, appointment_id: int) -> Appointment:
        # Fetch appointment data from database using appointment_id and db_util
        # Raise AppointmentNotFoundException if not found
        appointment_data = self.db_util.fetch_appointment_by_id(appointment_id)
        if not appointment_data:
            raise AppointmentNotFoundException(f"Appointment with ID {appointment_id} not found")
        return Appointment(**appointment_data)

    def get_appointments_for_patient(self, patient_id: int) -> List[Appointment]:
```

```
    # Retrieve appointments for the given patient_id using db_util
    appointment_data_list = self.db_util.fetch_appointments_for_patient(patient_id)
    return [Appointment(**data) for data in appointment_data_list]

def get_appointments_for_doctor(self, doctor_id: int) -> List[Appointment]:
    # ... (Similar implementation using db_util)
    pass

def schedule_appointment(self, appointment: Appointment) -> bool:
    # Add appointment to database using db_util
    return self.db_util.add_appointment(appointment)

def update_appointment(self, appointment: Appointment) -> bool:
    # Update appointment in database using db_util
    return self.db_util.update_appointment(appointment)

def cancel_appointment(self, appointment_id: int) -> bool:
    # Cancel appointment in database using db_util
    return self.db_util.cancel_appointment(appointment_id)
```