Dokumentasi syntax SQL

Pembuatan Database : CREATE DATABASE rumah_kita;

Pembuatan Tabel :

- Pengguna

```
rumah_kita=# CREATE TABLE pengguna (
rumah_kita(# id_pengguna SERIAL PRIMARY KEY,
rumah_kita(# nama VARCHAR(100) NOT NULL,
rumah_kita(# email VARCHAR(100) UNIQUE NOT NULL,
rumah_kita(# password VARCHAR(100),
rumah_kita(# created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP);
```

- Kamar

```
rumah_kita=# CREATE TABLE kamar (
rumah_kita(# id_kamar SERIAL PRIMARY KEY,
rumah_kita(# nama_kamar VARCHAR(100) NOT NULL,
rumah_kita(# harga_perbulan NUMERIC(10, 2) NOT NULL,
rumah_kita(# created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP);
```

- Booking

```
rumah_kita=# CREATE TABLE booking (
rumah_kita(# id_boking SERIAL PRIMARY KEY,
rumah_kita(# id_pengguna INT NOT NULL,
rumah_kita(# id_kamar INT NOT NULL,
rumah_kita(# tanggal_mulai_sewa DATE NOT NULL,
rumah_kita(# FOREIGN KEY (id_pengguna) REFERENCES
pengguna(id_pengguna) ON DELETE CASCADE,
rumah_kita(# FOREIGN KEY (id_kamar) REFERENCES kamar(id_kamar) ON
DELETE CASCADE);
```

Source Code Pemrograman dengan Python

```
from kivy.app import App
from kivy.uix.boxlayout import BoxLayout
from kivy.uix.button import Button
from kivy.uix.label import Label
from kivy.uix.textinput import TextInput
from kivy.uix.scrollview import ScrollView
from kivy.uix.gridlayout import GridLayout
from kivy.uix.popup import Popup
from sqlalchemy import create_engine, Column, Integer, String, Numeric, Date, TIMESTAMP,
ForeignKey, func
from sqlalchemy.ext.declarative import declarative_base
from sqlalchemy.orm import sessionmaker, relationship
import xlsxwriter
import psycopg2
import os
from datetime import date

Base = declarative_base()

# Database connection
def connect_db():
    return psycopg2.connect(
        dbname="rumah_kita",
        user="postgres",
        password="admin",
        host="localhost",
        port="5432"
    )

# Define models
class Pengguna(Base):
    __tablename__ = 'pengguna'

    id_pengguna = Column(Integer, primary_key=True, autoincrement=True)
    nama = Column(String(100), nullable=False)
    email = Column(String(100), unique=True, nullable=False)
    password = Column(String(100))
    created_at = Column(TIMESTAMP, default=func.current_timestamp())

class Kamar(Base):
    __tablename__ = 'kamar'

    id_kamar = Column(Integer, primary_key=True, autoincrement=True)
```

```
    nama_kamar = Column(String(100), nullable=False)
    harga_perbulan = Column(Numeric(10, 2), nullable=False)
    created_at = Column(TIMESTAMP, default=func.current_timestamp())

class Booking(Base):
    __tablename__ = 'booking'

    id_booking = Column(Integer, primary_key=True, autoincrement=True)
    id_pengguna = Column(Integer, ForeignKey('pengguna.id_pengguna', ondelete='CASCADE'),
nullable=False)
    id_kamar = Column(Integer, ForeignKey('kamar.id_kamar', ondelete='CASCADE'), nullable=False)
    tanggal_mulai_sewa = Column(Date, nullable=False)

# Database setup
connection = connect_db()
database_url = 'postgresql+psycopg2://postgres:admin@localhost:5432/rumah_kita'
engine = create_engine(database_url)
Base.metadata.create_all(engine)
Session = sessionmaker(bind=engine)
session = Session()

# Kivy App
class RumahKitaApp(App):
    def build(self):
        layout = BoxLayout(orientation='vertical')

        # Input fields untuk Pengguna
        self.nama_input = TextInput(hint_text='Nama', size_hint=(1, 0.1))
        self.email_input = TextInput(hint_text='Email', size_hint=(1, 0.1))
        self.password_input = TextInput(hint_text='Password', size_hint=(1, 0.1), password=True)
        add_pengguna_button = Button(text='Tambah Pengguna', size_hint=(1, 0.1))
        add_pengguna_button.bind(on_press=self.add_pengguna)

        layout.add_widget(self.nama_input)
        layout.add_widget(self.email_input)
        layout.add_widget(self.password_input)
        layout.add_widget(add_pengguna_button)

        # Input fields untuk Kamar
        self.nama_kamar_input = TextInput(hint_text='Nama Kamar', size_hint=(1, 0.1))
        self.harga_perbulan_input = TextInput(hint_text='Harga Perbulan', size_hint=(1, 0.1))
        add_kamar_button = Button(text='Tambah Kamar', size_hint=(1, 0.1))
        add_kamar_button.bind(on_press=self.add_kamar)
```

```
        layout.add_widget(self.nama_kamar_input)
        layout.add_widget(self.harga_perbulan_input)
        layout.add_widget(add_kamar_button)

        # Input fields untuk Booking
        self.id_pengguna_input = TextInput(hint_text='ID Pengguna', size_hint=(1, 0.1))
        self.id_kamar_input = TextInput(hint_text='ID Kamar', size_hint=(1, 0.1))
        add_booking_button = Button(text='Tambah Booking', size_hint=(1, 0.1))
        add_booking_button.bind(on_press=self.add_booking)

        layout.add_widget(self.id_pengguna_input)
        layout.add_widget(self.id_kamar_input)
        layout.add_widget(add_booking_button)

        # Export button
        export_button = Button(text='Export ke Excel', size_hint=(1, 0.1))
        export_button.bind(on_press=self.export_to_excel)
        layout.add_widget(export_button)

        # View data button
        view_data_button = Button(text='Lihat Semua Data', size_hint=(1, 0.1))
        view_data_button.bind(on_press=self.view_data)
        layout.add_widget(view_data_button)

        # Delete data button
        delete_data_button = Button(text='Hapus Data', size_hint=(1, 0.1))
        delete_data_button.bind(on_press=self.delete_data)
        layout.add_widget(delete_data_button)

        # Update data button
        update_data_button = Button(text='Update Data', size_hint=(1, 0.1))
        update_data_button.bind(on_press=self.update_data)
        layout.add_widget(update_data_button)

        return layout

    def add_pengguna(self, instance):
        nama = self.nama_input.text
        email = self.email_input.text
        password = self.password_input.text

        pengguna = Pengguna(nama=nama, email=email, password=password)
        session.add(pengguna)
        session.commit()
```

```
        self.nama_input.text = ''
        self.email_input.text = ''
        self.password_input.text = ''

        popup = Popup(title='Sukses', content=Label(text='Pengguna berhasil ditambahkan'), size_hint=(0.8,
0.4))
        popup.open()

    def add_kamar(self, instance):
        nama_kamar = self.nama_kamar_input.text
        harga_perbulan = self.harga_perbulan_input.text

        kamar = Kamar(nama_kamar=nama_kamar, harga_perbulan=harga_perbulan)
        session.add(kamar)
        session.commit()

        self.nama_kamar_input.text = ''
        self.harga_perbulan_input.text = ''

        popup = Popup(title='Sukses', content=Label(text='Kamar berhasil ditambahkan'), size_hint=(0.8,
0.4))
        popup.open()

    def add_booking(self, instance):
        id_pengguna = self.id_pengguna_input.text
        id_kamar = self.id_kamar_input.text
        tanggal_mulai_sewa = date.today()

        booking = Booking(id_pengguna=id_pengguna, id_kamar=id_kamar,
tanggal_mulai_sewa=tanggal_mulai_sewa)
        session.add(booking)
        session.commit()

        self.id_pengguna_input.text = ''
        self.id_kamar_input.text = ''

        popup = Popup(title='Sukses', content=Label(text='Booking berhasil ditambahkan'), size_hint=(0.8,
0.4))
        popup.open()

    def export_to_excel(self, instance):
        file_path = os.path.join(os.getcwd(), 'rumah_kita.xlsx')
```

```
with xlsxwriter.Workbook(file_path) as workbook:
    pengguna_sheet = workbook.add_worksheet('Pengguna')
    kamar_sheet = workbook.add_worksheet('Kamar')
    booking_sheet = workbook.add_worksheet('Booking')

    # Export Pengguna
    pengguna_sheet.write(0, 0, 'ID')
    pengguna_sheet.write(0, 1, 'Nama')
    pengguna_sheet.write(0, 2, 'Email')
    pengguna_sheet.write(0, 3, 'Password')
    pengguna_sheet.write(0, 4, 'Created At')

    for i, pengguna in enumerate(session.query(Pengguna).all(), start=1):
        pengguna_sheet.write(i, 0, pengguna.id_pengguna)
        pengguna_sheet.write(i, 1, pengguna.nama)
        pengguna_sheet.write(i, 2, pengguna.email)
        pengguna_sheet.write(i, 3, pengguna.password)
        pengguna_sheet.write(i, 4, str(pengguna.created_at))

    # Export Kamar
    kamar_sheet.write(0, 0, 'ID')
    kamar_sheet.write(0, 1, 'Nama Kamar')
    kamar_sheet.write(0, 2, 'Harga Perbulan')
    kamar_sheet.write(0, 3, 'Created At')

    for i, kamar in enumerate(session.query(Kamar).all(), start=1):
        kamar_sheet.write(i, 0, kamar.id_kamar)
        kamar_sheet.write(i, 1, kamar.nama_kamar)
        kamar_sheet.write(i, 2, float(kamar.harga_perbulan))
        kamar_sheet.write(i, 3, str(kamar.created_at))

    # Export Booking
    booking_sheet.write(0, 0, 'ID Booking')
    booking_sheet.write(0, 1, 'ID Pengguna')
    booking_sheet.write(0, 2, 'ID Kamar')
    booking_sheet.write(0, 3, 'Tanggal Mulai Sewa')

    for i, booking in enumerate(session.query(Booking).all(), start=1):
        booking_sheet.write(i, 0, booking.id_booking)
        booking_sheet.write(i, 1, booking.id_pengguna)
        booking_sheet.write(i, 2, booking.id_kamar)
        booking_sheet.write(i, 3, str(booking.tanggal_mulai_sewa))

popup = Popup(title='Sukses',
```

```
            content=Label(text=f'Data di ekspor ke {file_path}'),
            size_hint=(0.8, 0.4))
        popup.open()

    def view_data(self, instance):
        layout = GridLayout(cols=1, padding=10, spacing=10, size_hint_y=None)
        layout.bind(minimum_height=layout.setter('height'))

        # Pengguna data
        pengguna_label = Label(text='Pengguna Data', size_hint_y=None, height=40)
        layout.add_widget(pengguna_label)
        pengguna_data = GridLayout(cols=5, size_hint_y=None)
        pengguna_data.bind(minimum_height=pengguna_data.setter('height'))
        pengguna_data.add_widget(Label(text='ID'))
        pengguna_data.add_widget(Label(text='Nama'))
        pengguna_data.add_widget(Label(text='Email'))
        pengguna_data.add_widget(Label(text='Password'))
        pengguna_data.add_widget(Label(text='Created At'))

        for pengguna in session.query(Pengguna).all():
            pengguna_data.add_widget(Label(text=str(pengguna.id_pengguna)))
            pengguna_data.add_widget(Label(text=pengguna.nama))
            pengguna_data.add_widget(Label(text=pengguna.email))
            pengguna_data.add_widget(Label(text=pengguna.password))
            pengguna_data.add_widget(Label(text=str(pengguna.created_at)))

        layout.add_widget(pengguna_data)

        # Kamar data
        kamar_label = Label(text='Kamar Data', size_hint_y=None, height=40)
        layout.add_widget(kamar_label)
        kamar_data = GridLayout(cols=4, size_hint_y=None)
        kamar_data.bind(minimum_height=kamar_data.setter('height'))
        kamar_data.add_widget(Label(text='ID'))
        kamar_data.add_widget(Label(text='Nama Kamar'))
        kamar_data.add_widget(Label(text='Harga Perbulan'))
        kamar_data.add_widget(Label(text='Created At'))

        for kamar in session.query(Kamar).all():
            kamar_data.add_widget(Label(text=str(kamar.id_kamar)))
            kamar_data.add_widget(Label(text=kamar.nama_kamar))
            kamar_data.add_widget(Label(text=str(kamar.harga_perbulan)))
            kamar_data.add_widget(Label(text=str(kamar.created_at)))
```

```
        layout.add_widget(kamar_data)

        # Booking data
        booking_label = Label(text='Booking Data', size_hint_y=None, height=40)
        layout.add_widget(booking_label)
        booking_data = GridLayout(cols=4, size_hint_y=None)
        booking_data.bind(minimum_height=booking_data.setter('height'))
        booking_data.add_widget(Label(text='ID Booking'))
        booking_data.add_widget(Label(text='ID Pengguna'))
        booking_data.add_widget(Label(text='ID Kamar'))
        booking_data.add_widget(Label(text='Tanggal Mulai Sewa'))

        for booking in session.query(Booking).all():
            booking_data.add_widget(Label(text=str(booking.id_booking)))
            booking_data.add_widget(Label(text=str(booking.id_pengguna)))
            booking_data.add_widget(Label(text=str(booking.id_kamar)))
            booking_data.add_widget(Label(text=str(booking.tanggal_mulai_sewa)))

        layout.add_widget(booking_data)

        scroll_view = ScrollView(size_hint=(1, None), size=(self.root.width, self.root.height))
        scroll_view.add_widget(layout)

        popup = Popup(title='Lihat Data', content=scroll_view, size_hint=(0.9, 0.9))
        popup.open()

    # delete data
    def delete_data(self, instance):
        layout = BoxLayout(orientation='vertical', padding=10, spacing=10)

        self.delete_id_input = TextInput(hint_text='Masukan ID yang akan di hapus', size_hint=(1, 0.1))
        self.delete_table_input = TextInput(hint_text='Tabel (pengguna/kamar/booking)', size_hint=(1, 0.1))
        delete_button = Button(text='Hapus', size_hint=(1, 0.1))
        delete_button.bind(on_press=self.delete_record)

        layout.add_widget(self.delete_id_input)
        layout.add_widget(self.delete_table_input)
        layout.add_widget(delete_button)

        popup = Popup(title='Hapus Data', content=layout, size_hint=(0.8, 0.4))
        popup.open()

    def delete_record(self, instance):
        record_id = self.delete_id_input.text
```

```
        table_name = self.delete_table_input.text

        if table_name == 'pengguna':
            record = session.query(Pengguna).filter_by(id_pengguna=record_id).first()
        elif table_name == 'kamar':
            record = session.query(Kamar).filter_by(id_kamar=record_id).first()
        elif table_name == 'booking':
            record = session.query(Booking).filter_by(id_booking=record_id).first()
        else:
            popup = Popup(title='Error', content=Label(text='Nama tabel tidak valid'), size_hint=(0.8, 0.4))
            popup.open()
            return

        if record:
            session.delete(record)
            session.commit()
            popup = Popup(title='Sukses', content=Label(text='Berhasil dihapus'), size_hint=(0.8, 0.4))
        else:
            popup = Popup(title='Error', content=Label(text='data tidak ditemukan'), size_hint=(0.8, 0.4))

        popup.open()

    # update data
    def update_data(self, instance):
        layout = BoxLayout(orientation='vertical', padding=10, spacing=10)

        self.update_id_input = TextInput(hint_text='Masukan ID yang akan di update', size_hint=(1, 0.1))
        self.update_table_input = TextInput(hint_text='Tabel (pengguna/kamar/booking)', size_hint=(1, 0.1))
        self.update_field_input = TextInput(hint_text='masukan nama kolom yang akan di update',
size_hint=(1, 0.1))
        self.update_value_input = TextInput(hint_text='masukan nama baru untuk kolom yang akan di
update', size_hint=(1, 0.1))
        update_button = Button(text='Update', size_hint=(1, 0.1))
        update_button.bind(on_press=self.update_record)

        layout.add_widget(self.update_id_input)
        layout.add_widget(self.update_table_input)
        layout.add_widget(self.update_field_input)
        layout.add_widget(self.update_value_input)
        layout.add_widget(update_button)

        popup = Popup(title='Update Data', content=layout, size_hint=(0.8, 0.4))
        popup.open()
```

```
    def update_record(self, instance):
        record_id = self.update_id_input.text
        table_name = self.update_table_input.text
        field_name = self.update_field_input.text
        new_value = self.update_value_input.text

        if table_name == 'pengguna':
            record = session.query(Pengguna).filter_by(id_pengguna=record_id).first()
        elif table_name == 'kamar':
            record = session.query(Kamar).filter_by(id_kamar=record_id).first()
        elif table_name == 'booking':
            record = session.query(Booking).filter_by(id_booking=record_id).first()
        else:
            popup = Popup(title='Error', content=Label(text='Nama tabel tidak valid'), size_hint=(0.8, 0.4))
            popup.open()
            return

        if record:
            setattr(record, field_name, new_value)
            session.commit()
            popup = Popup(title='Sukses', content=Label(text='Update Berhasil'), size_hint=(0.8, 0.4))
        else:
            popup = Popup(title='Error', content=Label(text='Data tidak ditemukan'), size_hint=(0.8, 0.4))

        popup.open()

if __name__ == '__main__':
    RumahKitaApp().run()
```