**ASSIGNMENT\_3**

**NAME**: D. SAIMANIKANTA

**REG NO**: 192372348

**COURSE**:DATA BASE MANAGEMENT

SYSTEM

**CODE**:CSA0593

SCENERIO:

**Design a database for storing and processing data generated by smart home devices in real-time.Requirements:Model tables to handle devices, sensor readings, locations, and time-stamped data logs.Implement partitioning based on time intervals (e.g., daily) to efficiently manage large volumes of time-series data.Write queries to generate reports on device usage trends, detect anomalies in sensor readings, and visualize data by room or area.**

**Smart Home Data Storage and Processing System Design**

In designing a database for storing and processing data generated by smart home devices in real-time, we need to consider tables for managing devices, sensor readings, locations, and time-stamped data logs. We will also implement partitioning to optimize performance for time-series data and create queries to analyze device usage trends, detect anomalies, and visualize data by room or area.

**1. Database Schema Design**

The database will have the following entities:

1. **Devices**: Represents individual smart home devices.
2. **Sensors**: Stores information about different sensors attached to devices.
3. **Locations**: Represents the physical locations of devices (e.g., rooms, floors).
4. **Sensor Readings**: Contains data recorded from sensors, such as temperature, humidity, light intensity, etc.
5. **Data Logs**: Contains time-stamped logs of sensor readings for analysis.

**Table 1: Devices**

This table stores metadata about the smart home devices.

|  |
| --- |
| CREATE TABLE Devices (  device\_id SERIAL PRIMARY KEY,  device\_name VARCHAR(255),  device\_type VARCHAR(255), *-- e.g., "thermostat", "light", "door lock"*  status VARCHAR(50), *-- e.g., "active", "inactive"*  location\_id INT REFERENCES Locations(location\_id), *-- Foreign key to Locations table*  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP |

**Table 2: Locations**

This table stores details about the physical locations where devices are deployed (e.g., rooms, floors).

|  |
| --- |
| CREATE TABLE Locations (  location\_id SERIAL PRIMARY KEY,  location\_name VARCHAR(255), *-- e.g., "Living Room", "Kitchen"*  area VARCHAR(255), *-- e.g., "First Floor", "Second Floor"*  description TEXT  ); |

**Table 3: Sensors**

This table stores information about sensors attached to devices. A device can have multiple sensors.

|  |
| --- |
| CREATE TABLE Sensors (  sensor\_id SERIAL PRIMARY KEY,  device\_id INT REFERENCES Devices(device\_id), *-- Foreign key to Devices*  sensor\_type VARCHAR(255), *-- e.g., "temperature", "humidity", "motion"*  unit VARCHAR(50), *-- e.g., "°C", "percent", "lux"*  created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP  ); |

**Table 4: Sensor Readings**

This table stores individual sensor readings from devices. It captures data values such as temperature, humidity, etc.

|  |
| --- |
| CREATE TABLE SensorReadings (  reading\_id SERIAL PRIMARY KEY,  sensor\_id INT REFERENCES Sensors(sensor\_id), *-- Foreign key to Sensors*  timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  reading\_value DOUBLE PRECISION, *-- The actual value recorded by the sensor*  status VARCHAR(50) *-- e.g., "normal", "error"*  ); |

**Table 5: Data Logs**

This table stores time-stamped logs for sensor readings, potentially partitioned by date

|  |
| --- |
| CREATE TABLE DataLogs (  log\_id SERIAL PRIMARY KEY,  sensor\_reading\_id INT REFERENCES SensorReadings(reading\_id), *-- Foreign key to SensorReadings*  timestamp TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,  log\_message TEXT |

**2. Partitioning Strategy**

Given the time-series nature of the data, the SensorReadings and DataLogs tables will be partitioned based on time intervals (e.g., daily partitions). This partitioning improves performance by ensuring that only the relevant partitions are queried when accessing data for specific time ranges.

For example, to partition the SensorReadings table by day:

|  |
| --- |
| CREATE TABLE SensorReadings\_2024\_11\_29 PARTITION OF SensorReadings  FOR VALUES FROM ('2024-11-29') TO ('2024-11-30'); |

This partitioning ensures efficient querying, especially when querying for sensor data within a particular date range, reducing the data that needs to be scanned.

Similarly, partitioning can be applied to the DataLogs table to keep logs efficient and manageable:

**3. SQL Queries for Reporting and Analytics**

**1. Device Usage Trend Report**

To track device usage trends (e.g., the number of sensor readings per device per day), we can aggregate data by device\_id and timestamp.

|  |
| --- |
| SELECT device\_id,  DATE(timestamp) AS day,  COUNT(\*) AS readings\_count  FROM SensorReadings  GROUP BY device\_id, day  ORDER BY day DESC; |

This query will give us the number of sensor readings recorded for each device, grouped by day.

**2. Anomaly Detection Query**

Anomalies in sensor readings can be defined as values that exceed certain thresholds. For example, detecting extreme temperature values:

|  |
| --- |
| SELECT s.device\_id,  r.timestamp,  r.reading\_value,  s.sensor\_type  FROM SensorReadings r  JOIN Sensors s ON r.sensor\_id = s.sensor\_id  WHERE s.sensor\_type = 'temperature'  AND (r.reading\_value > 30 OR r.reading\_value < -5); |

This query provides the average temperature by room or area, which can be visualized on a dashboard.

**4. Device Performance Over Time**

To track the performance of devices over time, such as the number of active sensor readings, we can use:

|  |
| --- |
| SELECT d.device\_name,  COUNT(r.reading\_id) AS total\_readings,  DATE(r.timestamp) AS day  FROM SensorReadings r  JOIN Sensors s ON r.sensor\_id = s.sensor\_id  JOIN Devices d ON s.device\_id = d.device\_id  GROUP BY d.device\_name, day  ORDER BY day DESC; |

This query returns the total number of readings recorded for each device over time.

**4. Materialized Views for Performance Optimization**

To optimize performance for frequent queries (e.g., daily averages, location-based aggregates), we can create materialized views. Here’s an example:

|  |
| --- |
| CREATE MATERIALIZED VIEW daily\_device\_usage AS  SELECT device\_id,  DATE(timestamp) AS day,  COUNT(\*) AS readings\_count  FROM SensorReadings  GROUP BY device\_id, day; |

**Conclusion**

The system design includes the following key components:

* **Tables**: For devices, sensors, sensor readings, locations, and time-stamped data logs.
* **Partitioning**: By date to efficiently manage large volumes of time-series data.
* **Queries**: To generate reports on device usage trends, detect anomalies in sensor readings, and visualize data by room or area.
* **Materialized Views**: To improve the performance of frequent analytics queries.

This system design ensures efficient storage, real-time processing, and analysis of data generated by smart home devices while maintaining scalability and performance for large volumes of time-series data.