Project: Smart Home Energy Management System (SHEMS)

Team Members:

Vamshi Madineni (vm2496)

Sai Rajeev Koppuravuri (rk4305)

In the evolving landscape of energy consumption, the Smart Home Energy Management System (SHEMS) emerges as a pivotal solution aimed at empowering homeowners to make informed decisions about their energy usage. In this project, we delve into the foundational design of a relational database that forms the backbone of SHEMS, focusing on the storage and management of historical energy usage data.

Project Overview:

The SHEMS database serves as a central repository deployed by energy providers, allowing homeowners to connect and gain insights into their energy consumption patterns. Through this system, users can access past energy usage and cost data, understand the impact of various appliances and settings, and optimize their energy consumption.

Key Components of the SHEMS Database:

1. Customer Information:

- Capture essential customer details, including name and billing address.

- Allow customers to associate multiple service locations with their account, accommodating properties like primary residences, vacation homes, or rental properties.

- Store comprehensive service location information, including address, unit number, acquisition date, square footage, number of bedrooms, and occupants.

2. Device Enrollment:

- Enable customers to enroll smart devices, such as AC systems, dryers, lights, and refrigerators, at each service location.

- Include device attributes like type, model number, and pre-stored information about device properties.

- Assign unique identifiers to each enrolled device to ensure proper tracking.

3. Device Data Collection:

- Facilitate the collection of device-generated data, including events (on/off, setting changes, door status) and energy consumption information.

- Store data items with details such as device ID, timestamp, event labels, and corresponding values.

- Regularly capture energy consumption data, providing a comprehensive view of usage patterns.

4. Energy Prices:

- Track varying energy prices on an hourly basis, considering location-specific rates based on zip codes.

- Allow computation of energy costs at specific times, enabling users to understand the financial implications of their energy usage.

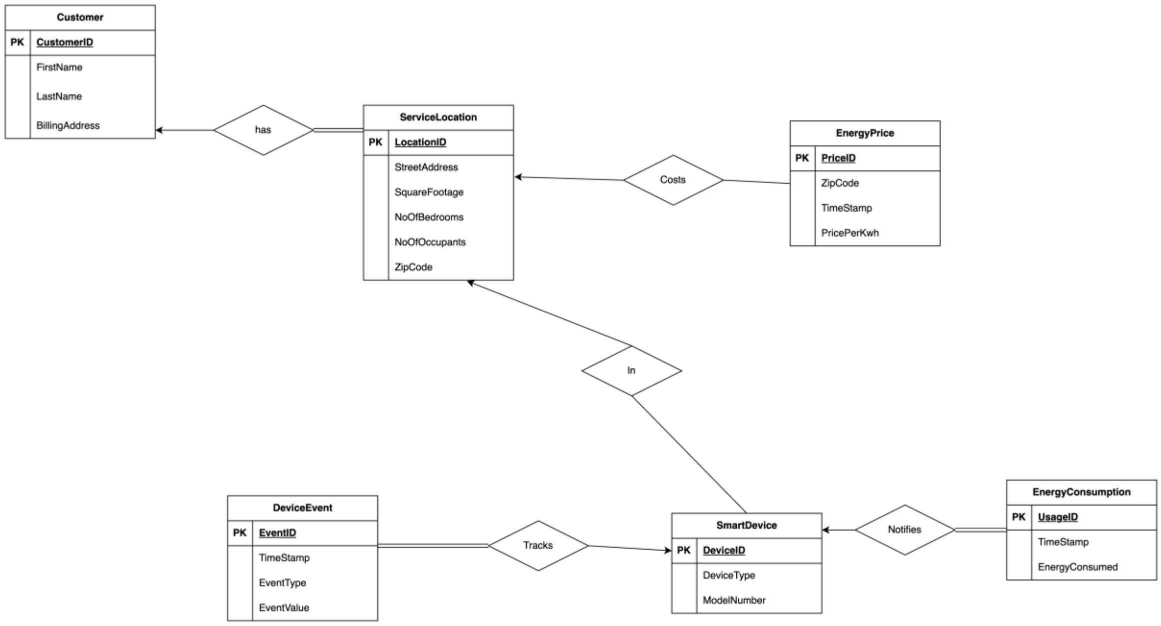
Project Goals:

- Design a robust relational schema to support the storage and retrieval of diverse data types related to customer information, service locations, device enrollment, and energy consumption.

- Establish clear relationships between entities, ensuring seamless navigation and retrieval of relevant data.

- Lay the foundation for the second part of the project, where a web-accessible frontend will be developed to provide users with a graphical interface for exploring their energy usage data.

ER Diagram:



The relationships between the entities are as follows:

A customer can have one or more service locations.

A service location has one energy price.

A smart device is installed at one service location.

A device event is generated by one smart device.

The ER diagram also shows the following attributes for each entity:

Entities:

- **Customer**
  - **Attributes:** CustomerID (Primary Key), FirstName, LastName, BillingAddress
  - **Justification:** Each customer information who subscribes to the SHEMS.
- **ServiceLocation**
  - **Attributes:** LocationID (Primary Key), StreetAddress, SquareFootage, NoOfBedrooms, NoOfOccupants, ZipCode, CustomerID (Foreign Key references Customer (CustomerID))

    - **Justification:** Each service location associated with a customer, facilitating energy usage comparisons.

- **SmartDevice**
    - **Attributes:** DeviceID (Primary Key), DeviceType, ModelNumber, LocationID (Foreign Key references ServiceLocation (LocationID))
    - **Justification:** Stores the devices enrolled by customers, including information about the device model.

- **DeviceEvent**
    - **Attributes:** EventID (Primary Key), DeviceID (Foreign Key), Timestamp, EventType, EventValue
    - **Justification:** Stores events generated by devices, such as switching on/off, changes in settings, door opening/closing, and energy consumption.

- **EnergyConsumption**
    - **Attributes:** UsageID (Primary Key), TimeStamp, EnergyConsumed, DeviceID (Foreign Key)
    - **Justification:** Storing information related to the energy consumption of enrolled devices.

- **EnergyPrice**
    - **Attributes:** PriceID, ZipCode, TimeStamp, PricePerKwh
    - **Justification:** Keeps track of energy prices over time, allowing cost calculations based on location and timestamp.

Assumptions:

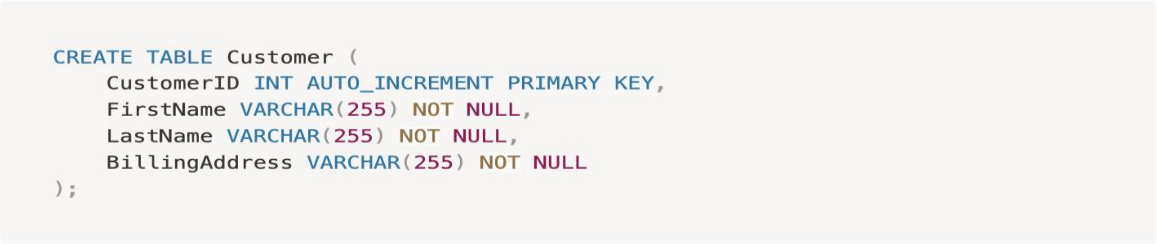
- Assumed that devices can be enrolled at any time, and historical data is retained.
- Timestamps in EnergyData and EnergyPrice are assumed to be in a standardized format.
- The schema assumes a simplified model for device settings and prestored information.
- Each Smart device has a unique DeviceID that differentiates multiple devices of similar type and model.

Normalization and Justification:

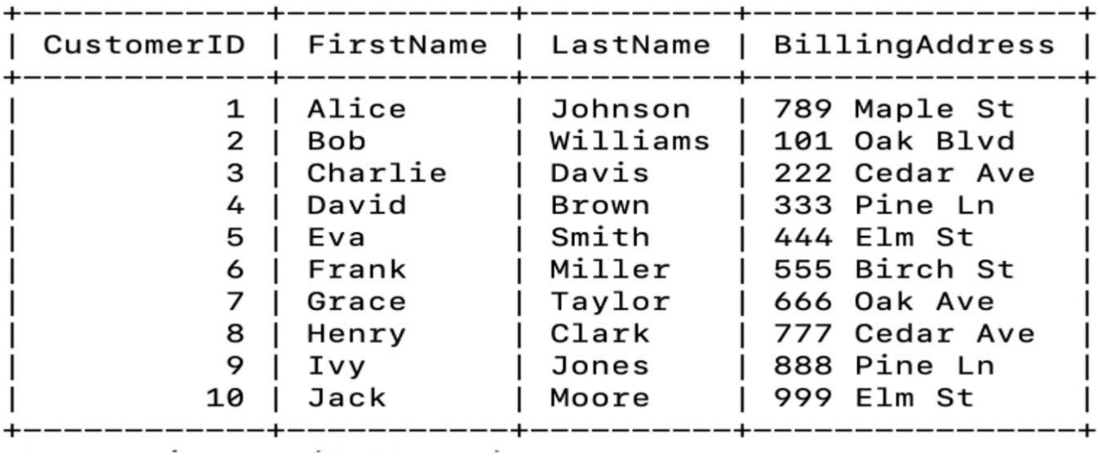
- The schema is normalized to eliminate redundancy and improve data integrity.
- Surrogate keys (CustomerID, LocationID, DeviceID, EventID, PriceID, UsageID) ensure unique identification.
- Foreign key constraints maintain referential integrity between tables.

Create Statements and Tabular Data:

Customer:

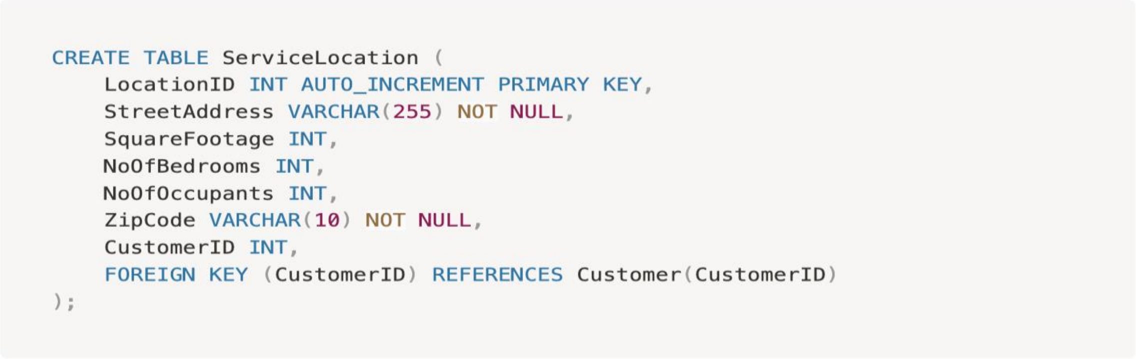


```sql
CREATE TABLE Customer (
    CustomerID INT AUTO_INCREMENT PRIMARY KEY,
    FirstName VARCHAR(255) NOT NULL,
    LastName VARCHAR(255) NOT NULL,
    BillingAddress VARCHAR(255) NOT NULL
);
```

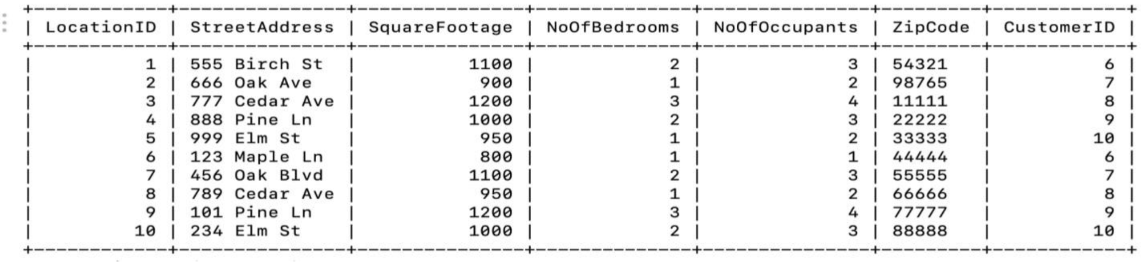


```
+------------+-----------+----------+----------------+
| CustomerID | FirstName | LastName | BillingAddress |
+------------+-----------+----------+----------------+
|          1 | Alice     | Johnson  | 789 Maple St   |
|          2 | Bob       | Williams | 101 Oak Blvd   |
|          3 | Charlie   | Davis    | 222 Cedar Ave  |
|          4 | David     | Brown    | 333 Pine Ln    |
|          5 | Eva       | Smith    | 444 Elm St     |
|          6 | Frank     | Miller   | 555 Birch St   |
|          7 | Grace     | Taylor   | 666 Oak Ave    |
|          8 | Henry     | Clark    | 777 Cedar Ave  |
|          9 | Ivy       | Jones    | 888 Pine Ln    |
|         10 | Jack      | Moore    | 999 Elm St     |
+------------+-----------+----------+----------------+
```

Service Location:

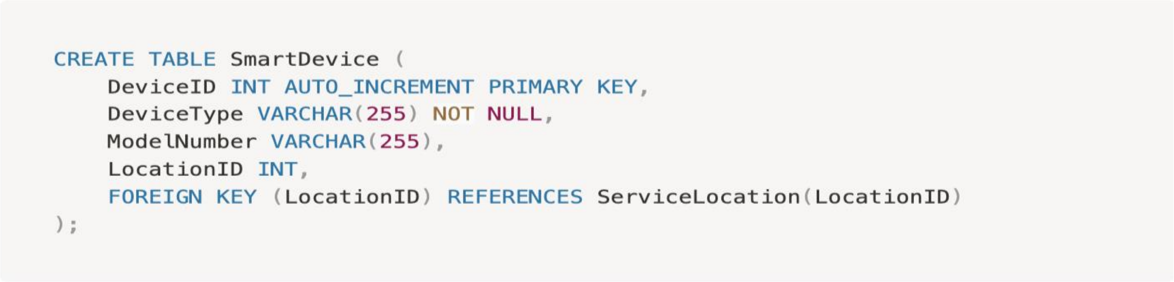


```
CREATE TABLE ServiceLocation (
    LocationID INT AUTO_INCREMENT PRIMARY KEY,
    StreetAddress VARCHAR(255) NOT NULL,
    SquareFootage INT,
    NoOfBedrooms INT,
    NoOfOccupants INT,
    ZipCode VARCHAR(10) NOT NULL,
    CustomerID INT,
    FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID)
);
```

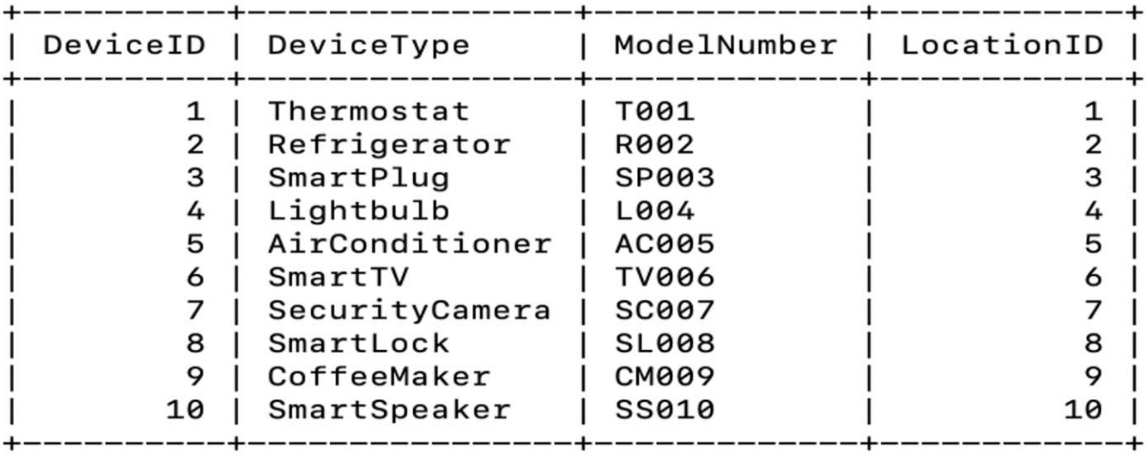


```
+------------+---------------+---------------+--------------+---------------+---------+------------+
| LocationID | StreetAddress | SquareFootage | NoOfBedrooms | NoOfOccupants | ZipCode | CustomerID |
+------------+---------------+---------------+--------------+---------------+---------+------------+
|          1 | 555 Birch St  |          1100 |            2 |             3 | 54321   |          6 |
|          2 | 666 Oak Ave   |           900 |            1 |             2 | 98765   |          7 |
|          3 | 777 Cedar Ave |          1200 |            3 |             4 | 11111   |          8 |
|          4 | 888 Pine Ln   |          1000 |            2 |             3 | 22222   |          9 |
|          5 | 999 Elm St    |           950 |            1 |             2 | 33333   |         10 |
|          6 | 123 Maple Ln  |           800 |            1 |             1 | 44444   |          6 |
|          7 | 456 Oak Blvd  |          1100 |            2 |             3 | 55555   |          7 |
|          8 | 789 Cedar Ave |           950 |            1 |             2 | 66666   |          8 |
|          9 | 101 Pine Ln   |          1200 |            3 |             4 | 77777   |          9 |
|         10 | 234 Elm St    |          1000 |            2 |             3 | 88888   |         10 |
+------------+---------------+---------------+--------------+---------------+---------+------------+
```

Smart Device:

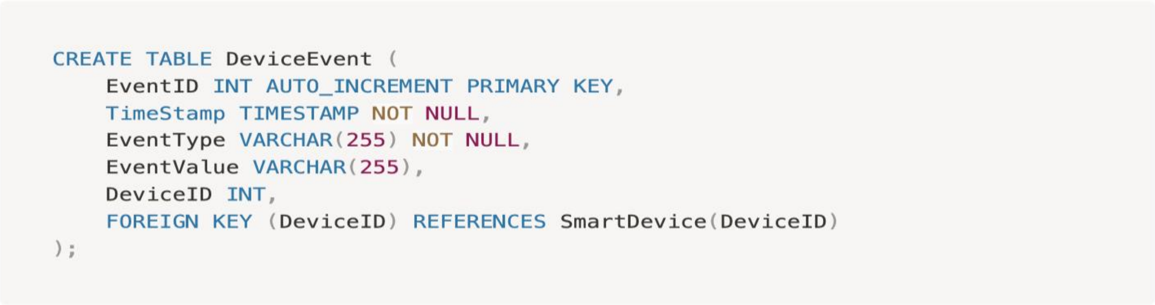


```
CREATE TABLE SmartDevice (
    DeviceID INT AUTO_INCREMENT PRIMARY KEY,
    DeviceType VARCHAR(255) NOT NULL,
    ModelNumber VARCHAR(255),
    LocationID INT,
    FOREIGN KEY (LocationID) REFERENCES ServiceLocation(LocationID)
);
```

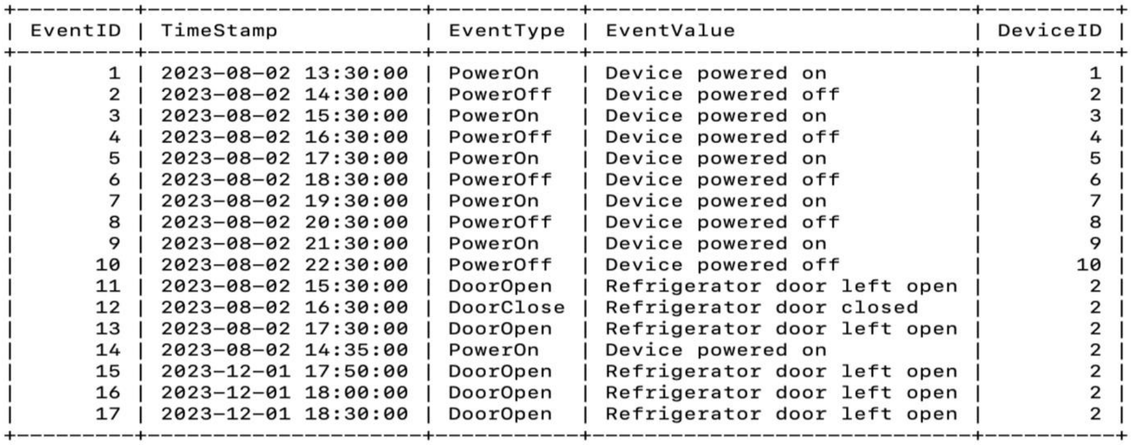


```
+----------+----------------+-------------+------------+
| DeviceID | DeviceType     | ModelNumber | LocationID |
+----------+----------------+-------------+------------+
|        1 | Thermostat     | T001        |          1 |
|        2 | Refrigerator   | R002        |          2 |
|        3 | SmartPlug      | SP003       |          3 |
|        4 | Lightbulb      | L004        |          4 |
|        5 | AirConditioner | AC005       |          5 |
|        6 | SmartTV        | TV006       |          6 |
|        7 | SecurityCamera | SC007       |          7 |
|        8 | SmartLock      | SL008       |          8 |
|        9 | CoffeeMaker    | CM009       |          9 |
|       10 | SmartSpeaker   | SS010       |         10 |
+----------+----------------+-------------+------------+
```

Device Event:

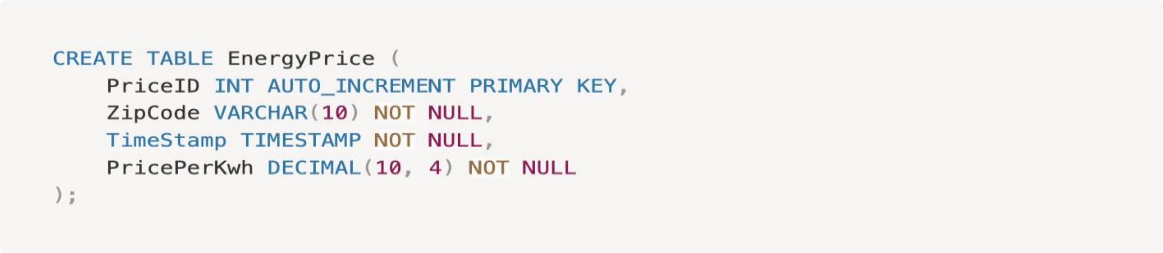


```sql
CREATE TABLE DeviceEvent (
    EventID INT AUTO_INCREMENT PRIMARY KEY,
    TimeStamp TIMESTAMP NOT NULL,
    EventType VARCHAR(255) NOT NULL,
    EventValue VARCHAR(255),
    DeviceID INT,
    FOREIGN KEY (DeviceID) REFERENCES SmartDevice(DeviceID)
);
```

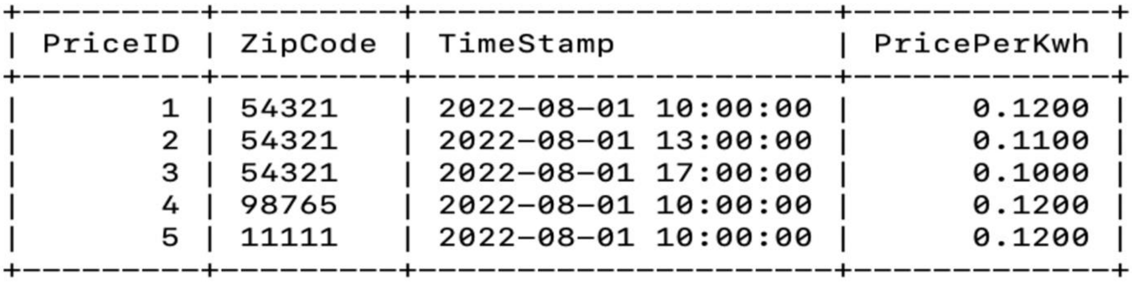


```
+---------+---------------------+-----------+-----------------------------+----------+
| EventID | TimeStamp           | EventType | EventValue                  | DeviceID |
+---------+---------------------+-----------+-----------------------------+----------+
|       1 | 2023-08-02 13:30:00 | PowerOn   | Device powered on           |        1 |
|       2 | 2023-08-02 14:30:00 | PowerOff  | Device powered off          |        2 |
|       3 | 2023-08-02 15:30:00 | PowerOn   | Device powered on           |        3 |
|       4 | 2023-08-02 16:30:00 | PowerOff  | Device powered off          |        4 |
|       5 | 2023-08-02 17:30:00 | PowerOn   | Device powered on           |        5 |
|       6 | 2023-08-02 18:30:00 | PowerOff  | Device powered off          |        6 |
|       7 | 2023-08-02 19:30:00 | PowerOn   | Device powered on           |        7 |
|       8 | 2023-08-02 20:30:00 | PowerOff  | Device powered off          |        8 |
|       9 | 2023-08-02 21:30:00 | PowerOn   | Device powered on           |        9 |
|      10 | 2023-08-02 22:30:00 | PowerOff  | Device powered off          |       10 |
|      11 | 2023-08-02 15:30:00 | DoorOpen  | Refrigerator door left open |        2 |
|      12 | 2023-08-02 16:30:00 | DoorClose | Refrigerator door closed    |        2 |
|      13 | 2023-08-02 17:30:00 | DoorOpen  | Refrigerator door left open |        2 |
|      14 | 2023-08-02 14:35:00 | PowerOn   | Device powered on           |        2 |
|      15 | 2023-12-01 17:50:00 | DoorOpen  | Refrigerator door left open |        2 |
|      16 | 2023-12-01 18:00:00 | DoorOpen  | Refrigerator door left open |        2 |
|      17 | 2023-12-01 18:30:00 | DoorOpen  | Refrigerator door left open |        2 |
+---------+---------------------+-----------+-----------------------------+----------+
```

Energy Price:

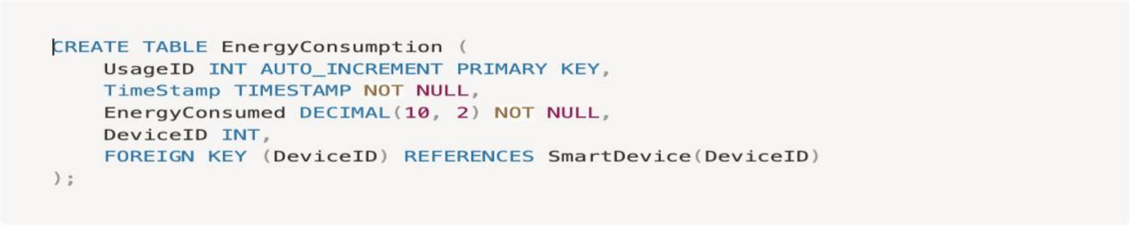


```sql
CREATE TABLE EnergyPrice (
    PriceID INT AUTO_INCREMENT PRIMARY KEY,
    ZipCode VARCHAR(10) NOT NULL,
    TimeStamp TIMESTAMP NOT NULL,
    PricePerKwh DECIMAL(10, 4) NOT NULL
);
```

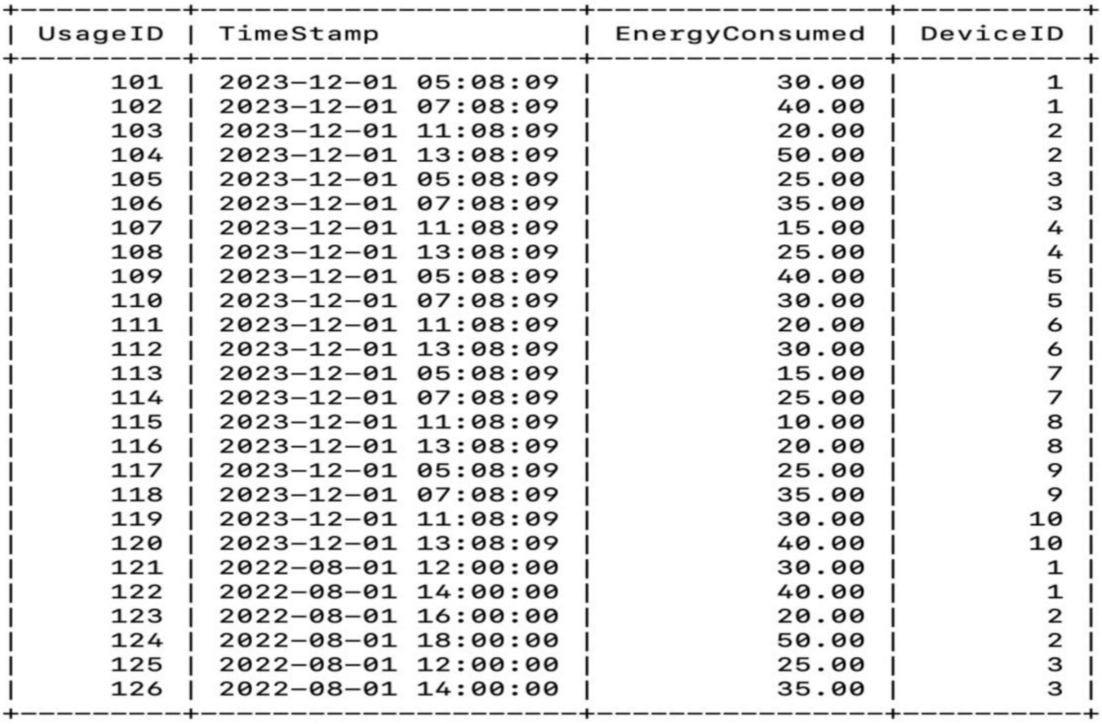


```
+---------+---------+---------------------+-------------+
| PriceID | ZipCode | TimeStamp           | PricePerKwh |
+---------+---------+---------------------+-------------+
|       1 | 54321   | 2022-08-01 10:00:00 |      0.1200 |
|       2 | 54321   | 2022-08-01 13:00:00 |      0.1100 |
|       3 | 54321   | 2022-08-01 17:00:00 |      0.1000 |
|       4 | 98765   | 2022-08-01 10:00:00 |      0.1200 |
|       5 | 11111   | 2022-08-01 10:00:00 |      0.1200 |
+---------+---------+---------------------+-------------+
```

Energy Consumption:



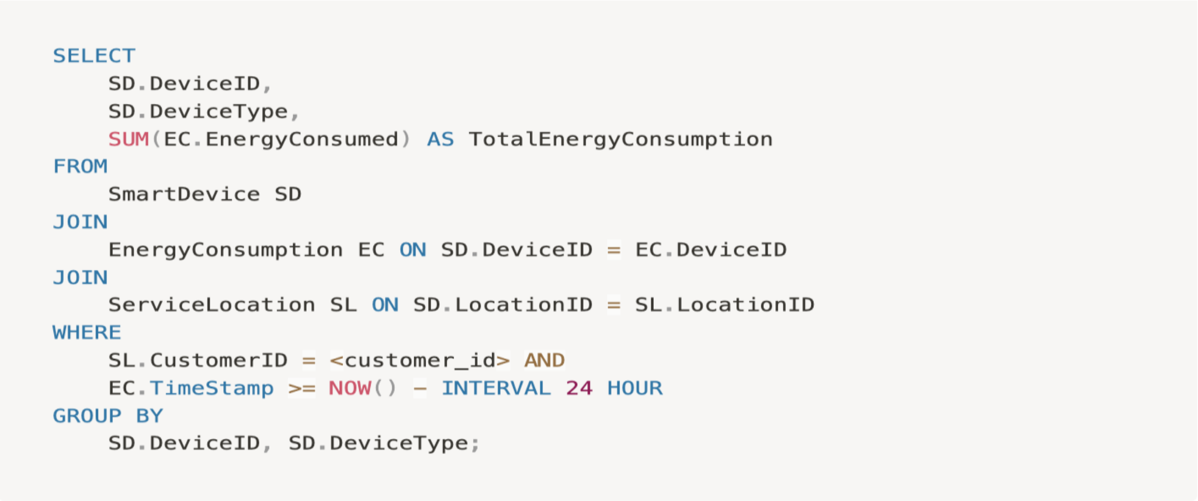
```
CREATE TABLE EnergyConsumption (
    UsageID INT AUTO_INCREMENT PRIMARY KEY,
    TimeStamp TIMESTAMP NOT NULL,
    EnergyConsumed DECIMAL(10, 2) NOT NULL,
    DeviceID INT,
    FOREIGN KEY (DeviceID) REFERENCES SmartDevice(DeviceID)
);
```



```
+----------+---------------------+----------------+----------+
| UsageID  | TimeStamp           | EnergyConsumed | DeviceID |
+----------+---------------------+----------------+----------+
|      101 | 2023-12-01 05:08:09 |          30.00 |        1 |
|      102 | 2023-12-01 07:08:09 |          40.00 |        1 |
|      103 | 2023-12-01 11:08:09 |          20.00 |        2 |
|      104 | 2023-12-01 13:08:09 |          50.00 |        2 |
|      105 | 2023-12-01 05:08:09 |          25.00 |        3 |
|      106 | 2023-12-01 07:08:09 |          35.00 |        3 |
|      107 | 2023-12-01 11:08:09 |          15.00 |        4 |
|      108 | 2023-12-01 13:08:09 |          25.00 |        4 |
|      109 | 2023-12-01 05:08:09 |          40.00 |        5 |
|      110 | 2023-12-01 07:08:09 |          30.00 |        5 |
|      111 | 2023-12-01 11:08:09 |          20.00 |        6 |
|      112 | 2023-12-01 13:08:09 |          30.00 |        6 |
|      113 | 2023-12-01 05:08:09 |          15.00 |        7 |
|      114 | 2023-12-01 07:08:09 |          25.00 |        7 |
|      115 | 2023-12-01 11:08:09 |          10.00 |        8 |
|      116 | 2023-12-01 13:08:09 |          20.00 |        8 |
|      117 | 2023-12-01 05:08:09 |          25.00 |        9 |
|      118 | 2023-12-01 07:08:09 |          35.00 |        9 |
|      119 | 2023-12-01 11:08:09 |          30.00 |       10 |
|      120 | 2023-12-01 13:08:09 |          40.00 |       10 |
|      121 | 2022-08-01 12:00:00 |          30.00 |        1 |
|      122 | 2022-08-01 14:00:00 |          40.00 |        1 |
|      123 | 2022-08-01 16:00:00 |          20.00 |        2 |
|      124 | 2022-08-01 18:00:00 |          50.00 |        2 |
|      125 | 2022-08-01 12:00:00 |          25.00 |        3 |
|      126 | 2022-08-01 14:00:00 |          35.00 |        3 |
+----------+---------------------+----------------+----------+
```

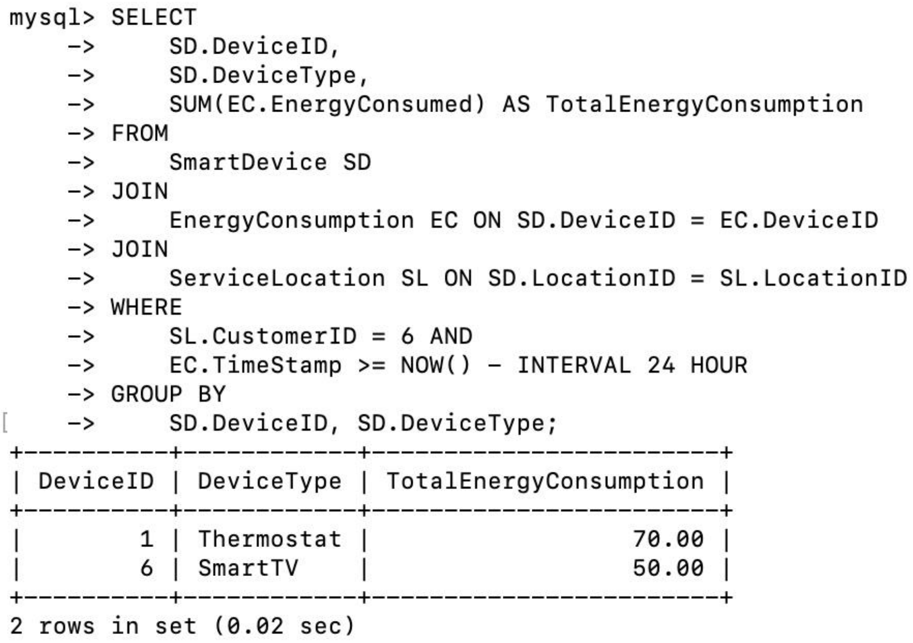
Queries & Output:

1. List all enrolled devices with their total energy consumption in the last 24 hours, for a specific customer identified by customer ID.



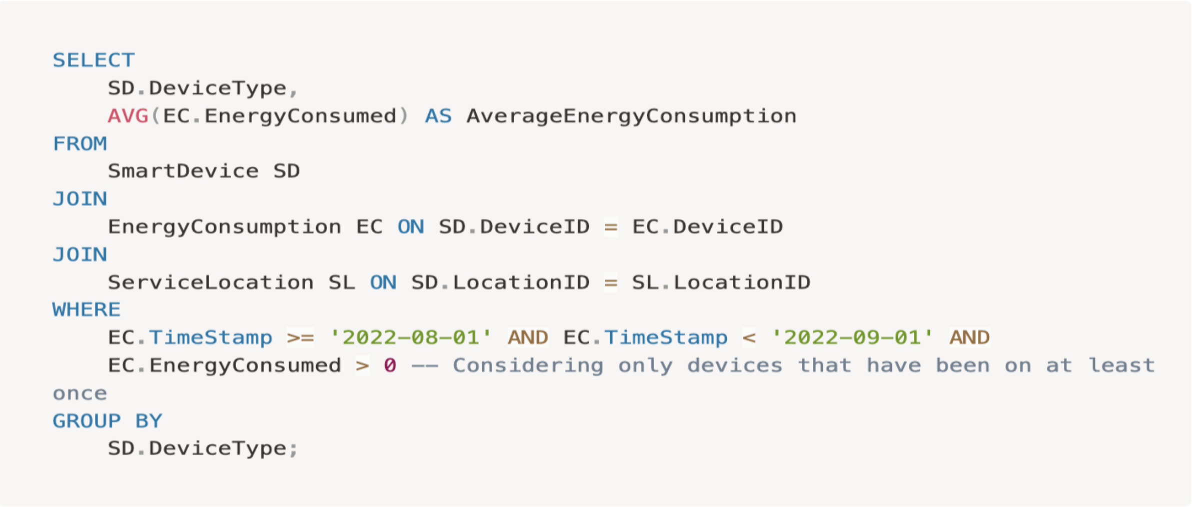
```
SELECT
    SD.DeviceID,
    SD.DeviceType,
    SUM(EC.EnergyConsumed) AS TotalEnergyConsumption
FROM
    SmartDevice SD
JOIN
    EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
JOIN
    ServiceLocation SL ON SD.LocationID = SL.LocationID
WHERE
    SL.CustomerID = <customer_id> AND
    EC.TimeStamp >= NOW() - INTERVAL 24 HOUR
GROUP BY
    SD.DeviceID, SD.DeviceType;
```

MYSQL Output:



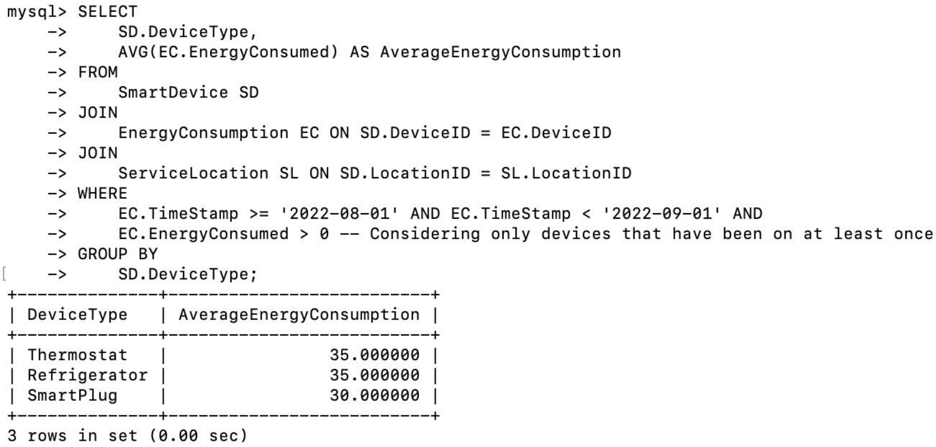
```
mysql> SELECT
    ->     SD.DeviceID,
    ->     SD.DeviceType,
    ->     SUM(EC.EnergyConsumed) AS TotalEnergyConsumption
    -> FROM
    ->     SmartDevice SD
    -> JOIN
    ->     EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    -> JOIN
    ->     ServiceLocation SL ON SD.LocationID = SL.LocationID
    -> WHERE
    ->     SL.CustomerID = 6 AND
    ->     EC.TimeStamp >= NOW() - INTERVAL 24 HOUR
    -> GROUP BY
    ->     SD.DeviceID, SD.DeviceType;
+----------+------------+------------------------+
| DeviceID | DeviceType | TotalEnergyConsumption |
+----------+------------+------------------------+
|        1 | Thermostat |                  70.00 |
|        6 | SmartTV    |                  50.00 |
+----------+------------+------------------------+
2 rows in set (0.02 sec)
```

2. Calculate the average monthly energy consumption per device type, for the month of August 2022, considering only devices that have been on (i.e., reported data) at least once during that month.



```
SELECT
    SD.DeviceType,
    AVG(EC.EnergyConsumed) AS AverageEnergyConsumption
FROM
    SmartDevice SD
JOIN
    EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
JOIN
    ServiceLocation SL ON SD.LocationID = SL.LocationID
WHERE
    EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01' AND
    EC.EnergyConsumed > 0 -- Considering only devices that have been on at least
once
GROUP BY
    SD.DeviceType;
```

MySQL Output:

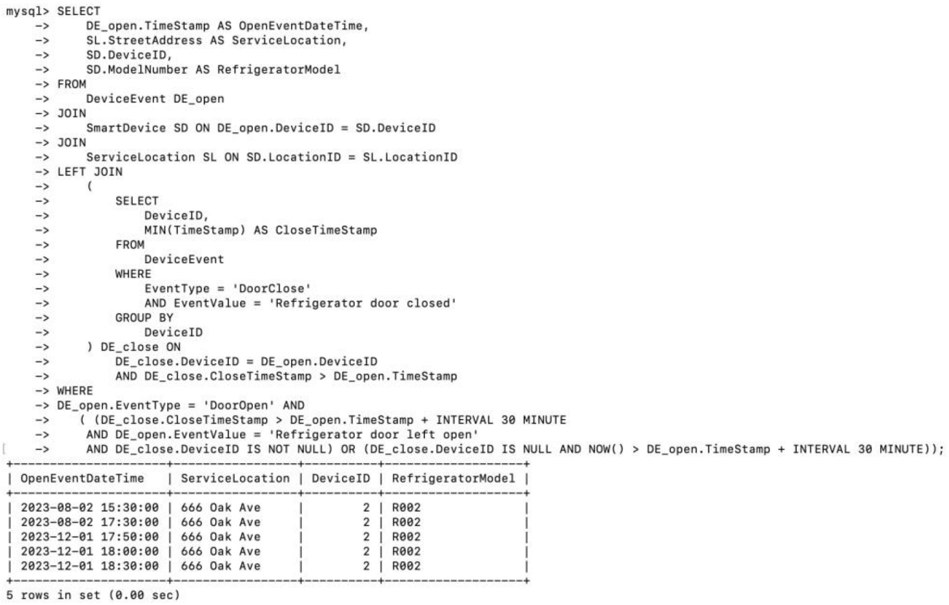


```
mysql> SELECT
    ->     SD.DeviceType,
    ->     AVG(EC.EnergyConsumed) AS AverageEnergyConsumption
    -> FROM
    ->     SmartDevice SD
    -> JOIN
    ->     EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    -> JOIN
    ->     ServiceLocation SL ON SD.LocationID = SL.LocationID
    -> WHERE
    ->     EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01' AND
    ->     EC.EnergyConsumed > 0 -- Considering only devices that have been on at least once
    -> GROUP BY
    ->     SD.DeviceType;
+--------------+--------------------------+
| DeviceType   | AverageEnergyConsumption |
+--------------+--------------------------+
| Thermostat   |                35.000000 |
| Refrigerator |                35.000000 |
| SmartPlug    |                30.000000 |
+--------------+--------------------------+
3 rows in set (0.00 sec)
```

3. Identify cases where a refrigerator door was left open for more than 30 minutes. Output the date and time, the service location, the device ID, and the refrigerator model.

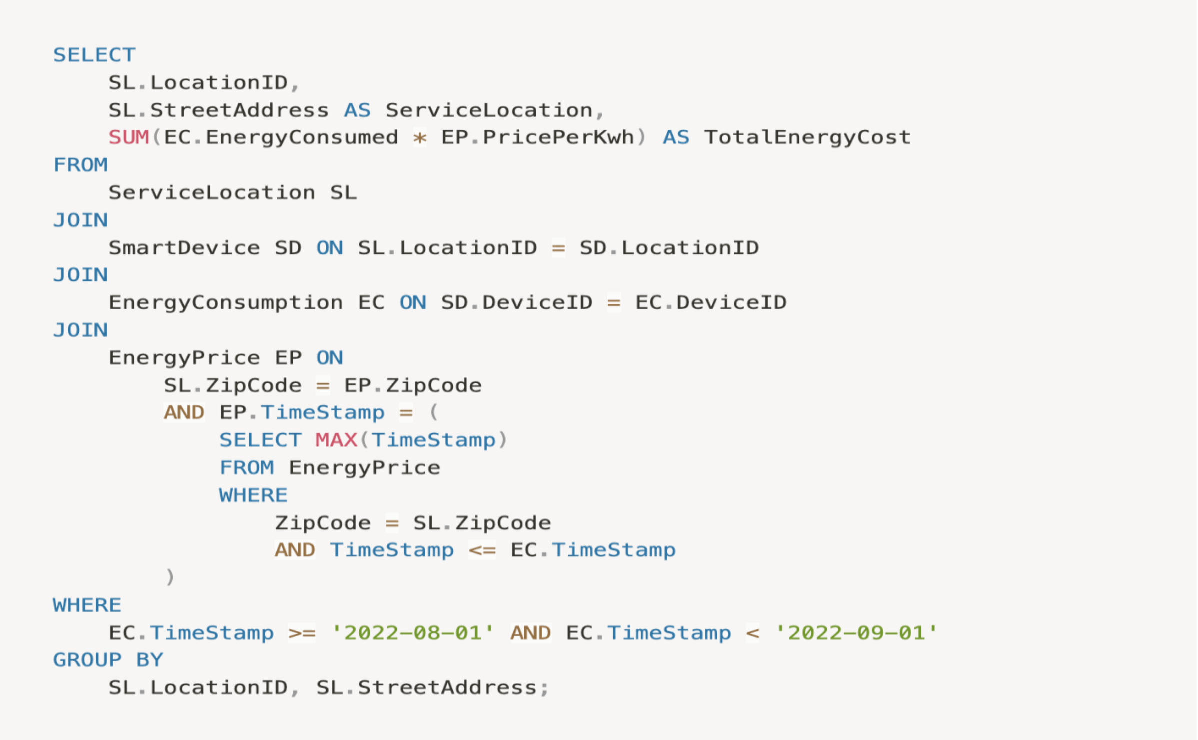
```
SELECT
    DE_open.TimeStamp AS OpenEventDateTime,
    SL.StreetAddress AS ServiceLocation,
    SD.DeviceID,
    SD.ModelNumber AS RefrigeratorModel
FROM
    DeviceEvent DE_open
JOIN
    SmartDevice SD ON DE_open.DeviceID = SD.DeviceID
JOIN
    ServiceLocation SL ON SD.LocationID = SL.LocationID
LEFT JOIN
    (
        SELECT
            DeviceID,
            MIN(TimeStamp) AS CloseTimeStamp
        FROM
            DeviceEvent
        WHERE
            EventType = 'DoorClose'
            AND EventValue = 'Refrigerator door closed'
        GROUP BY
            DeviceID
    ) DE_close ON
        DE_close.DeviceID = DE_open.DeviceID
        AND DE_close.CloseTimeStamp > DE_open.TimeStamp
WHERE
DE_open.EventType = 'DoorOpen' AND
   ( (DE_close.CloseTimeStamp > DE_open.TimeStamp + INTERVAL 30 MINUTE
    AND DE_open.EventValue = 'Refrigerator door left open'
    AND DE_close.DeviceID IS NOT NULL) OR (DE_close.DeviceID IS NULL AND NOW() >
DE_open.TimeStamp + INTERVAL 30 MINUTE));
```

MySQL Output:



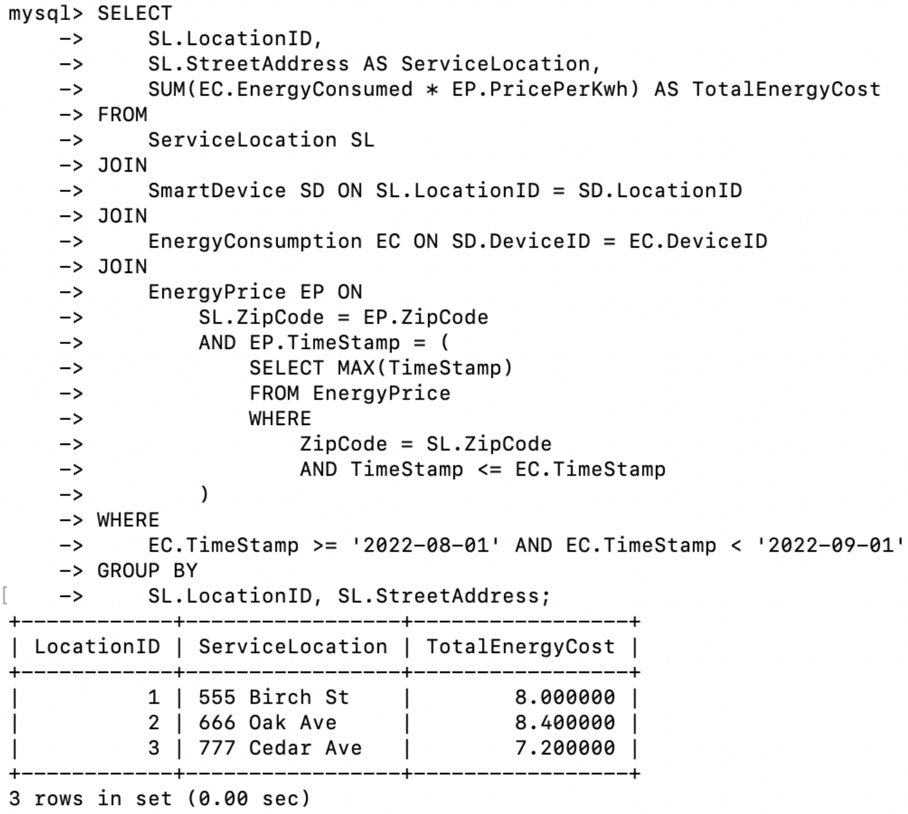
```
mysql> SELECT
    ->     DE_open.TimeStamp AS OpenEventDateTime,
    ->     SL.StreetAddress AS ServiceLocation,
    ->     SD.DeviceID,
    ->     SD.ModelNumber AS RefrigeratorModel
    -> FROM
    ->     DeviceEvent DE_open
    -> JOIN
    ->     SmartDevice SD ON DE_open.DeviceID = SD.DeviceID
    -> JOIN
    ->     ServiceLocation SL ON SD.LocationID = SL.LocationID
    -> LEFT JOIN
    ->     (
    ->         SELECT
    ->             DeviceID,
    ->             MIN(TimeStamp) AS CloseTimeStamp
    ->         FROM
    ->             DeviceEvent
    ->         WHERE
    ->             EventType = 'DoorClose'
    ->             AND EventValue = 'Refrigerator door closed'
    ->         GROUP BY
    ->             DeviceID
    ->     ) DE_close ON
    ->         DE_close.DeviceID = DE_open.DeviceID
    ->         AND DE_close.CloseTimeStamp > DE_open.TimeStamp
    -> WHERE
    -> DE_open.EventType = 'DoorOpen' AND
    ->    ( (DE_close.CloseTimeStamp > DE_open.TimeStamp + INTERVAL 30 MINUTE
    ->     AND DE_open.EventValue = 'Refrigerator door left open'
    ->     AND DE_close.DeviceID IS NOT NULL) OR (DE_close.DeviceID IS NULL AND NOW() > DE_open.TimeStamp + INTERVAL 30 MINUTE));
+---------------------+-----------------+----------+-------------------+
| OpenEventDateTime   | ServiceLocation | DeviceID | RefrigeratorModel |
+---------------------+-----------------+----------+-------------------+
| 2023-08-02 15:30:00 | 666 Oak Ave     |        2 | R002              |
| 2023-08-02 17:30:00 | 666 Oak Ave     |        2 | R002              |
| 2023-12-01 17:50:00 | 666 Oak Ave     |        2 | R002              |
| 2023-12-01 18:00:00 | 666 Oak Ave     |        2 | R002              |
| 2023-12-01 18:30:00 | 666 Oak Ave     |        2 | R002              |
+---------------------+-----------------+----------+-------------------+
5 rows in set (0.00 sec)
```

4. Calculate the total energy cost for each service location during August 2022, considering the hourly changing energy prices based on zip code.



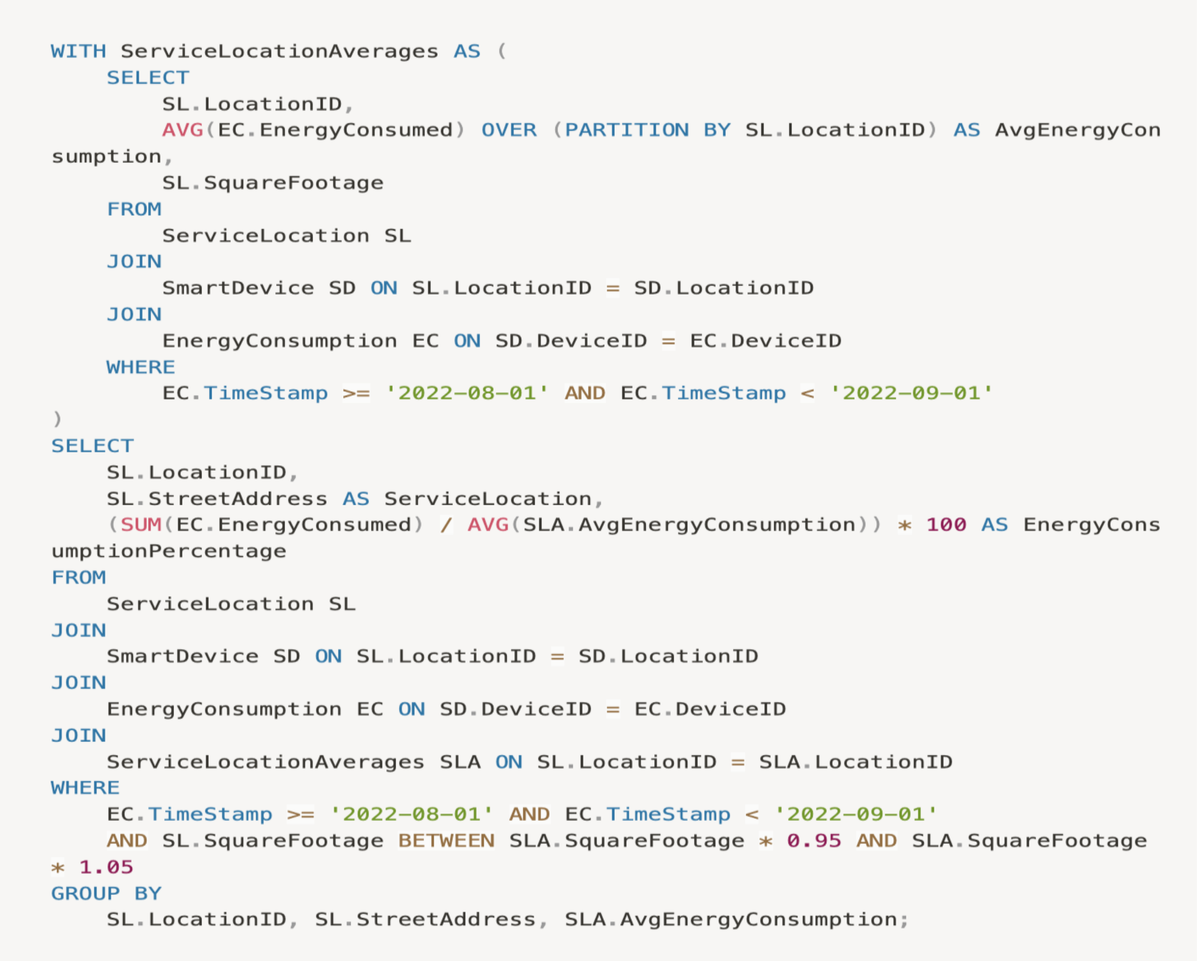
```sql
SELECT
    SL.LocationID,
    SL.StreetAddress AS ServiceLocation,
    SUM(EC.EnergyConsumed * EP.PricePerKwh) AS TotalEnergyCost
FROM
    ServiceLocation SL
JOIN
    SmartDevice SD ON SL.LocationID = SD.LocationID
JOIN
    EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
JOIN
    EnergyPrice EP ON
        SL.ZipCode = EP.ZipCode
        AND EP.TimeStamp = (
            SELECT MAX(TimeStamp)
            FROM EnergyPrice
            WHERE
                ZipCode = SL.ZipCode
                AND TimeStamp <= EC.TimeStamp
        )
WHERE
    EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
GROUP BY
    SL.LocationID, SL.StreetAddress;
```

MySQL Output:



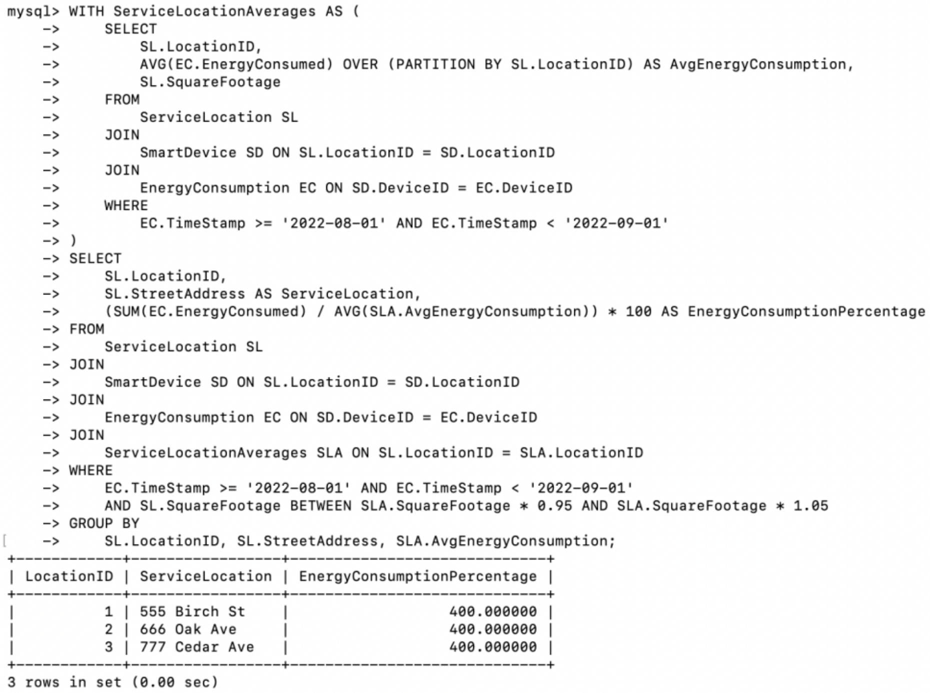
```
mysql> SELECT
    ->     SL.LocationID,
    ->     SL.StreetAddress AS ServiceLocation,
    ->     SUM(EC.EnergyConsumed * EP.PricePerKwh) AS TotalEnergyCost
    -> FROM
    ->     ServiceLocation SL
    -> JOIN
    ->     SmartDevice SD ON SL.LocationID = SD.LocationID
    -> JOIN
    ->     EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    -> JOIN
    ->     EnergyPrice EP ON
    ->         SL.ZipCode = EP.ZipCode
    ->         AND EP.TimeStamp = (
    ->             SELECT MAX(TimeStamp)
    ->             FROM EnergyPrice
    ->             WHERE
    ->                 ZipCode = SL.ZipCode
    ->                 AND TimeStamp <= EC.TimeStamp
    ->         )
    -> WHERE
    ->     EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
    -> GROUP BY
    ->     SL.LocationID, SL.StreetAddress;
+------------+-----------------+-----------------+
| LocationID | ServiceLocation | TotalEnergyCost |
+------------+-----------------+-----------------+
|          1 | 555 Birch St    |        8.000000 |
|          2 | 666 Oak Ave     |        8.400000 |
|          3 | 777 Cedar Ave   |        7.200000 |
+------------+-----------------+-----------------+
3 rows in set (0.00 sec)
```

5. For each service location, compute its total energy consumption during August 2022, as a percentage of the average total energy consumption during the same time of other service locations that have a similar square footage (meaning, at most 5% higher or lower square footage). Thus, you would output 150% if a service location with 1000 sqft had 50% higher energy consumption than the average of other service locations that have between 950 and 1050 sqft.



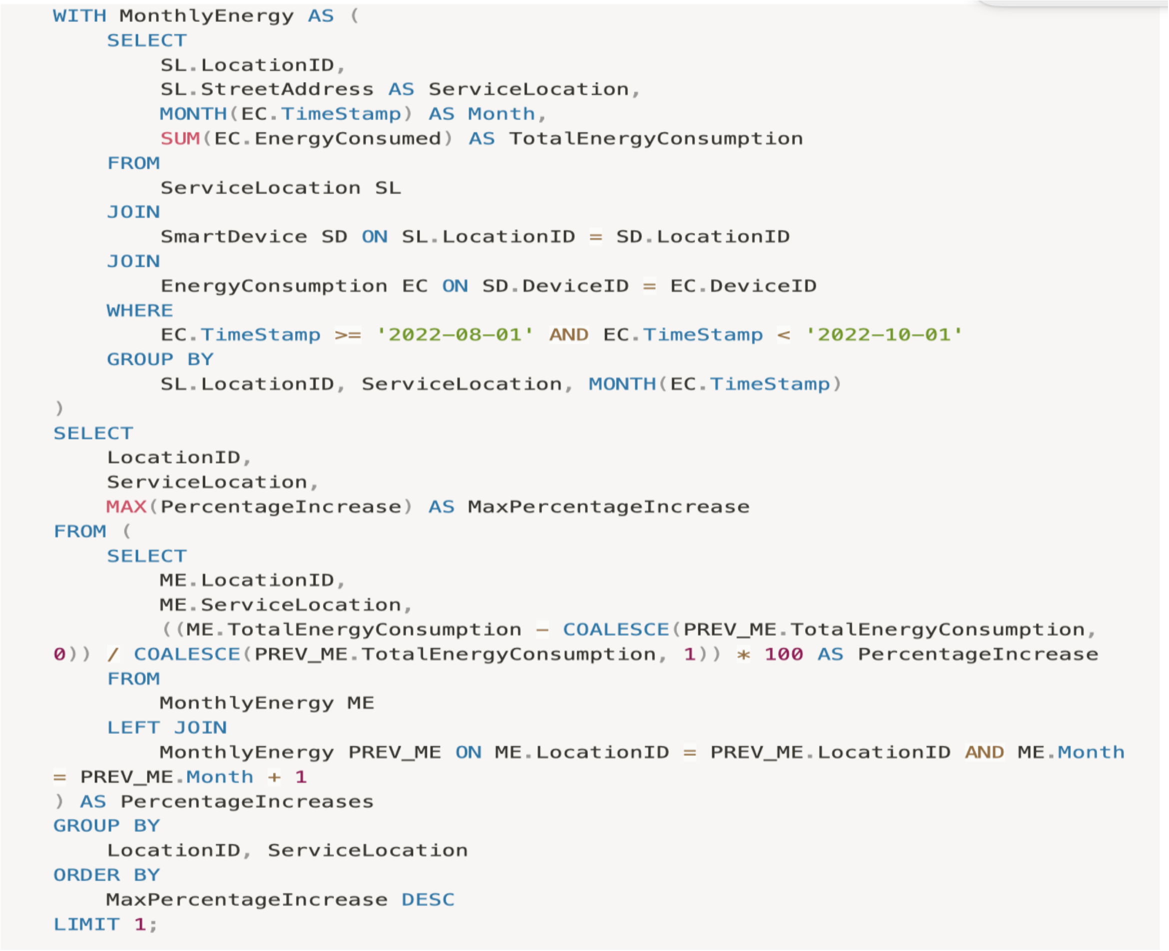
```sql
WITH ServiceLocationAverages AS (
    SELECT
        SL.LocationID,
        AVG(EC.EnergyConsumed) OVER (PARTITION BY SL.LocationID) AS AvgEnergyConsumption,
        SL.SquareFootage
    FROM
        ServiceLocation SL
    JOIN
        SmartDevice SD ON SL.LocationID = SD.LocationID
    JOIN
        EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    WHERE
        EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
)
SELECT
    SL.LocationID,
    SL.StreetAddress AS ServiceLocation,
    (SUM(EC.EnergyConsumed) / AVG(SLA.AvgEnergyConsumption)) * 100 AS EnergyConsumptionPercentage
FROM
    ServiceLocation SL
JOIN
    SmartDevice SD ON SL.LocationID = SD.LocationID
JOIN
    EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
JOIN
    ServiceLocationAverages SLA ON SL.LocationID = SLA.LocationID
WHERE
    EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
    AND SL.SquareFootage BETWEEN SLA.SquareFootage * 0.95 AND SLA.SquareFootage * 1.05
GROUP BY
    SL.LocationID, SL.StreetAddress, SLA.AvgEnergyConsumption;
```

MySQL Output:



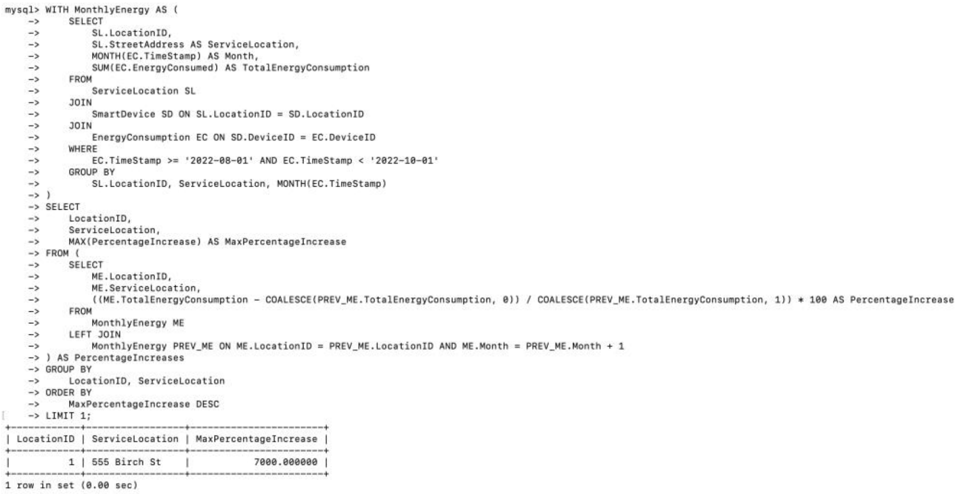
```
mysql> WITH ServiceLocationAverages AS (
    ->     SELECT
    ->         SL.LocationID,
    ->         AVG(EC.EnergyConsumed) OVER (PARTITION BY SL.LocationID) AS AvgEnergyConsumption,
    ->         SL.SquareFootage
    ->     FROM
    ->         ServiceLocation SL
    ->     JOIN
    ->         SmartDevice SD ON SL.LocationID = SD.LocationID
    ->     JOIN
    ->         EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    ->     WHERE
    ->         EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
    -> )
    -> SELECT
    ->     SL.LocationID,
    ->     SL.StreetAddress AS ServiceLocation,
    ->     (SUM(EC.EnergyConsumed) / AVG(SLA.AvgEnergyConsumption)) * 100 AS EnergyConsumptionPercentage
    -> FROM
    ->     ServiceLocation SL
    -> JOIN
    ->     SmartDevice SD ON SL.LocationID = SD.LocationID
    -> JOIN
    ->     EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    -> JOIN
    ->     ServiceLocationAverages SLA ON SL.LocationID = SLA.LocationID
    -> WHERE
    ->     EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-09-01'
    ->     AND SL.SquareFootage BETWEEN SLA.SquareFootage * 0.95 AND SLA.SquareFootage * 1.05
    -> GROUP BY
    ->     SL.LocationID, SL.StreetAddress, SLA.AvgEnergyConsumption;
+------------+-----------------+-----------------------------+
| LocationID | ServiceLocation | EnergyConsumptionPercentage |
+------------+-----------------+-----------------------------+
|          1 | 555 Birch St    |                  400.000000 |
|          2 | 666 Oak Ave     |                  400.000000 |
|          3 | 777 Cedar Ave   |                  400.000000 |
+------------+-----------------+-----------------------------+
3 rows in set (0.00 sec)
```

6. Identify service location(s) that had the highest percentage increase in energy consumption between August and September of 2022.



```
WITH MonthlyEnergy AS (
    SELECT
        SL.LocationID,
        SL.StreetAddress AS ServiceLocation,
        MONTH(EC.TimeStamp) AS Month,
        SUM(EC.EnergyConsumed) AS TotalEnergyConsumption
    FROM
        ServiceLocation SL
    JOIN
        SmartDevice SD ON SL.LocationID = SD.LocationID
    JOIN
        EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    WHERE
        EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-10-01'
    GROUP BY
        SL.LocationID, ServiceLocation, MONTH(EC.TimeStamp)
)
SELECT
    LocationID,
    ServiceLocation,
    MAX(PercentageIncrease) AS MaxPercentageIncrease
FROM (
    SELECT
        ME.LocationID,
        ME.ServiceLocation,
        ((ME.TotalEnergyConsumption - COALESCE(PREV_ME.TotalEnergyConsumption,
0)) / COALESCE(PREV_ME.TotalEnergyConsumption, 1)) * 100 AS PercentageIncrease
    FROM
        MonthlyEnergy ME
    LEFT JOIN
        MonthlyEnergy PREV_ME ON ME.LocationID = PREV_ME.LocationID AND ME.Month
= PREV_ME.Month + 1
) AS PercentageIncreases
GROUP BY
    LocationID, ServiceLocation
ORDER BY
    MaxPercentageIncrease DESC
LIMIT 1;
```

MySQL Output:



```
mysql> WITH MonthlyEnergy AS (
    ->     SELECT
    ->         SL.LocationID,
    ->         SL.StreetAddress AS ServiceLocation,
    ->         MONTH(EC.TimeStamp) AS Month,
    ->         SUM(EC.EnergyConsumed) AS TotalEnergyConsumption
    ->     FROM
    ->         ServiceLocation SL
    ->     JOIN
    ->         SmartDevice SD ON SL.LocationID = SD.LocationID
    ->     JOIN
    ->         EnergyConsumption EC ON SD.DeviceID = EC.DeviceID
    ->     WHERE
    ->         EC.TimeStamp >= '2022-08-01' AND EC.TimeStamp < '2022-10-01'
    ->     GROUP BY
    ->         SL.LocationID, ServiceLocation, MONTH(EC.TimeStamp)
    -> )
    -> SELECT
    ->     LocationID,
    ->     ServiceLocation,
    ->     MAX(PercentageIncrease) AS MaxPercentageIncrease
    -> FROM (
    ->     SELECT
    ->         ME.LocationID,
    ->         ME.ServiceLocation,
    ->         ((ME.TotalEnergyConsumption - COALESCE(PREV_ME.TotalEnergyConsumption, 0)) / COALESCE(PREV_ME.TotalEnergyConsumption, 1)) * 100 AS PercentageIncrease
    ->     FROM
    ->         MonthlyEnergy ME
    ->     LEFT JOIN
    ->         MonthlyEnergy PREV_ME ON ME.LocationID = PREV_ME.LocationID AND ME.Month = PREV_ME.Month + 1
    -> ) AS PercentageIncreases
    -> GROUP BY
    ->     LocationID, ServiceLocation
    -> ORDER BY
    ->     MaxPercentageIncrease DESC
    -> LIMIT 1;
+------------+-----------------+-----------------------+
| LocationID | ServiceLocation | MaxPercentageIncrease |
+------------+-----------------+-----------------------+
|          1 | 555 Birch St    |           7000.000000 |
+------------+-----------------+-----------------------+
1 row in set (0.00 sec)
```