```
function calculateAverageRent(inputLatitude, inputLongitude, maxDistance, yearBuiltLower, yearBuiltUpper, unitsLower, unitsUpper) {
  var projectId = 'rent-automation-project'; // Replace with your Google Cloud project ID
  var sqlQuery = `
    SELECT AVG(CAST('Studio Asking Rent-Unit' AS FLOAT64)) as averageRent
    FROM rental_properties.properties_data
    WHERE ST_DISTANCE(ST_GEOGPOINT(CAST(Longitude AS FLOAT64), CAST(Latitude AS FLOAT64)), ST_GEOGPOINT(${inputLongitude}, ${inputLatitude})) <= ${maxDistance * 1609.34}
    AND CAST('Year Built' AS INT64) >= ${yearBuiltLower}
    AND CAST('Year Built' AS INT64) <= ${yearBuiltUpper}
    AND CAST('Number of Units' AS INT64) >= ${unitsLower}
    AND CAST('Number of Units' AS INT64) <= ${unitsUpper}
  `;
  var queryRequest = {
    query: sqlQuery,
    useLegacySql: false
  };
  var queryResults = BigQuery.Jobs.query(queryRequest, projectId);
  var rows = queryResults.rows;

  return rows.length > 0 ? parseFloat(rows[0].f[0].v) : 0;
}
```

```javascript
function calculatePropertyCount(inputLatitude, inputLongitude, maxDistance, yearBuiltLower, yearBuiltUpper, unitsLower, unitsUpper) {
  var ss = SpreadsheetApp.getActiveSpreadsheet();
  var databaseSheet = ss.getSheetByName("Database");

  // Get data from 'Database' sheet
  var databaseData = databaseSheet.getDataRange().getValues();

  var propertyCount = 0; // Counter for properties that meet all criteria

  // Calculate distances and filter by maxDistance, Year Built, and Number of Units
  for (var i = 1; i < databaseData.length; i++) {
    var row = databaseData[i];
    var latitude = row[7]; // Assuming latitude is in column H
    var longitude = row[8]; // Assuming longitude is in column I
    var yearBuilt = row[1]; // Assuming year built is in column B
    var numberOfUnits = row[9]; // Assuming number of units is in column J

    var dist = calculateHaversineDistance(inputLatitude, inputLongitude, latitude, longitude);

    // Check distance, year built, and number of units criteria
    if (dist <= maxDistance &&
        yearBuilt >= yearBuiltLower && yearBuilt <= yearBuiltUpper &&
        numberOfUnits >= unitsLower && numberOfUnits <= unitsUpper) {
      propertyCount++;
    }
  }

  // Return the count of properties
  return propertyCount;
}

function calculateHaversineDistance(lat1, lon1, lat2, lon2) {
  var R = 6371; // Radius of the earth in km
  var dLat = degreesToRadians(lat2 - lat1);
  var dLon = degreesToRadians(lon2 - lon1);
  var a =
    Math.sin(dLat / 2) * Math.sin(dLat / 2) +
    Math.cos(degreesToRadians(lat1)) * Math.cos(degreesToRadians(lat2)) *
    Math.sin(dLon / 2) * Math.sin(dLon / 2);
  var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1 - a));
  var distance = R * c; // Distance in km
  return distance * 0.621371; // Convert to miles
}

function degreesToRadians(degrees) {
  return degrees * (Math.PI / 180);
```