Property Management Application

By **Marthand Bhargav**

Phase 8: Data Management & Deployment

Salesforce Rental Management System - Professional Implementation Guide

---

Table of Contents

---

1. Introduction

1.1 Purpose of Data Management & Deployment

Data Management and Deployment represent critical phases in any Salesforce implementation. These processes ensure that your carefully designed rental management system transitions smoothly from development to production while maintaining data integrity and system reliability.

Key Objectives:

- Migrate sample and production data into Salesforce objects
- Maintain data quality through validation and duplicate management
- Deploy custom metadata, code, and configurations across environments
- Establish repeatable deployment processes for future updates
- Create backup and recovery strategies for business continuity

1.2 Importance of Salesforce Tools

Salesforce provides a comprehensive toolkit for data and metadata management:

Data Import Wizard: Ideal for one-time imports of up to 50,000 records with a user-friendly interface. Perfect for initial setup of Rental Units and Agreements.

Data Loader: Command-line and GUI tool supporting bulk operations (insert, update, upsert, delete) for millions of records. Essential for large-scale Statement and Payment imports.

Change Sets: Point-and-click deployment tool for moving customizations between related Salesforce environments. Suitable for deploying Rental objects, Apex classes, and Aura components.

Salesforce DX (SFDX): Modern development paradigm enabling source-driven development, version control integration, and automated deployments. Represents the future of Salesforce development.

ANT Migration Tool: Java-based deployment tool using metadata API. Enables scripted, automated deployments ideal for CI/CD pipelines.

These tools work together to create a robust ecosystem for managing your rental management system throughout its lifecycle.

---

2. Data Import & Management

2.1 Data Import Wizard

The Data Import Wizard is the most accessible tool for importing data into Salesforce. It's ideal for initial data loads and smaller datasets.

2.1.1 Importing Rental Units

Step 1: Prepare Your Data File

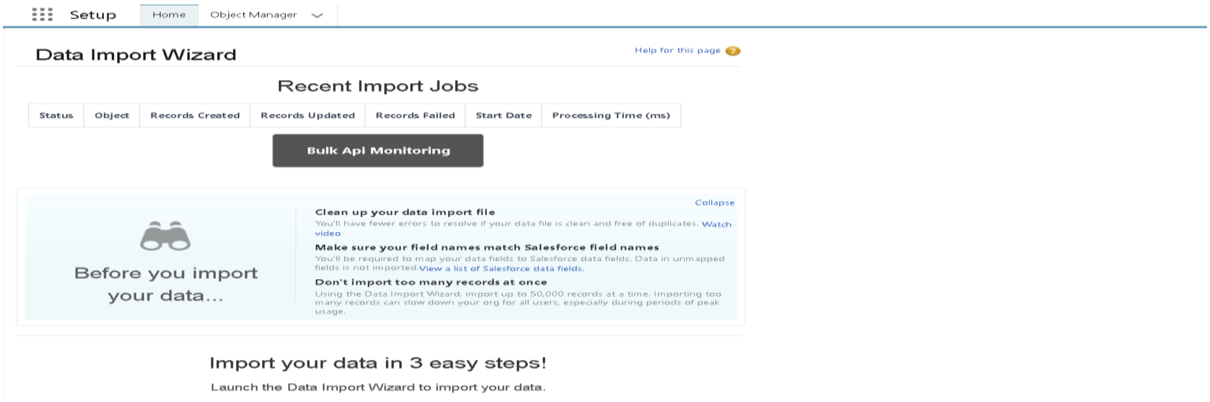
Create a CSV file containing your rental unit data with the following columns:

- Unit_Number__c
- Property_Address__c

- Property_City__c
- Property_State__c
- Property_ZIP__c
- Number_of_Bedrooms__c
- Number_of_Bathrooms__c
- Monthly_Rent__c
- Security_Deposit__c
- Unit_Status__c (Available, Occupied, Maintenance, Vacant)
- Property_Type__c (Apartment, House, Condo, Townhouse)
- Date_Available__c (format: YYYY-MM-DD)

Step 2: Access Data Import Wizard

1. Navigate to **Setup** (gear icon in top right)
2. In the Quick Find box, type "Data Import Wizard"
3. Click **Data Import Wizard**
4. Click **Launch Wizard**



Recent Import Jobs

Status	Object	Records Created	Records Updated	Records Failed	Start Date	Processing Time (ms)

Clean up your data import file
You'll have fewer errors to resolve if your data file is clean and free of duplicates. Watch video

Make sure your field names match Salesforce field names
You'll be required to map your data fields to Salesforce data fields. Data in unmapped fields is not imported.View a list of Salesforce data fields.

Don't import too many records at once
Using the Data Import Wizard, import up to 50,000 records at a time. Importing too many records can slow down your org for all users, especially during periods of peak usage.

Import your data in 3 easy steps!

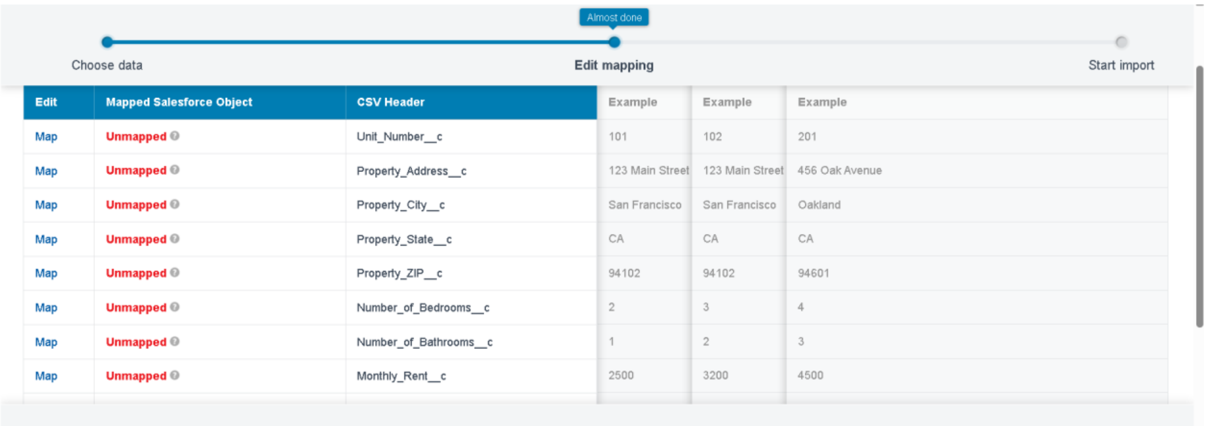
Launch the Data Import Wizard to import your data.

Step 3: Configure Import Settings

1. Select **Custom objects** radio button
2. From the dropdown, select **Rental Units**
3. Choose **Add new records**
4. Click **Next**

Step 4: Upload Your CSV File

1. Click **Choose File** and select your prepared CSV
2. Select **Comma** as the field delimiter
3. Select **UTF-8** as character encoding
4. Click **Next**



Edit	Mapped Salesforce Object	CSV Header	Example	Example	Example
Map	Unmapped	Unit_Number__c	101	102	201
Map	Unmapped	Property_Address__c	123 Main Street	123 Main Street	456 Oak Avenue
Map	Unmapped	Property_City__c	San Francisco	San Francisco	Oakland
Map	Unmapped	Property_State__c	CA	CA	CA
Map	Unmapped	Property_ZIP__c	94102	94102	94601
Map	Unmapped	Number_of_Bedrooms__c	2	3	4
Map	Unmapped	Number_of_Bathrooms__c	1	2	3
Map	Unmapped	Monthly_Rent__c	2500	3200	4500

Step 5: Map Fields

1. The wizard will attempt to auto-map fields based on column headers
2. Review each mapping carefully:
    - Unit_Number__c → Unit Number
    - Property_Address__c → Property Address
    - Monthly_Rent__c → Monthly Rent
3. For any unmapped fields, manually select the correct Salesforce field from dropdown
4. Click **Next**

Step 6: Review and Start Import

1. Review the import summary
2. Click **Start Import**
3. You'll receive an email when the import completes

4. Click **OK** to close the wizard

2.1.2 Importing Rental Agreements

Step 1: Prepare Rental Agreement CSV

Before importing Rental Agreements, note that you need:

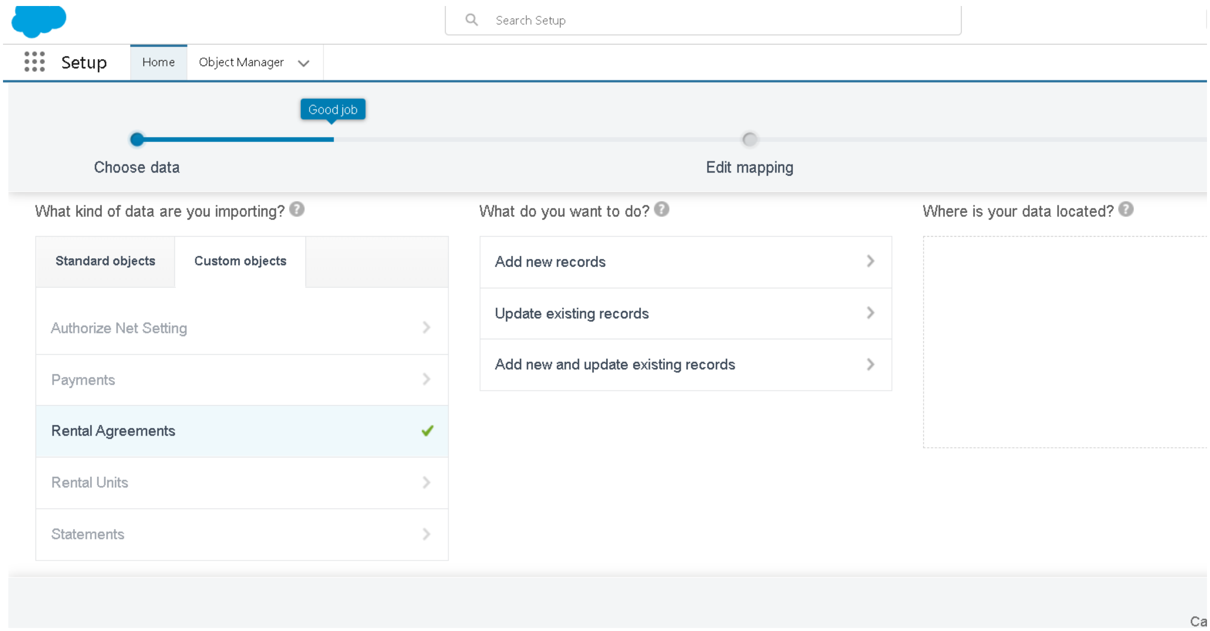
- **Rental_Unit__c**: The Salesforce ID of the related Rental Unit (obtained after importing units)
- **Renter__c**: The Salesforce ID of the Contact record

Example CSV structure:

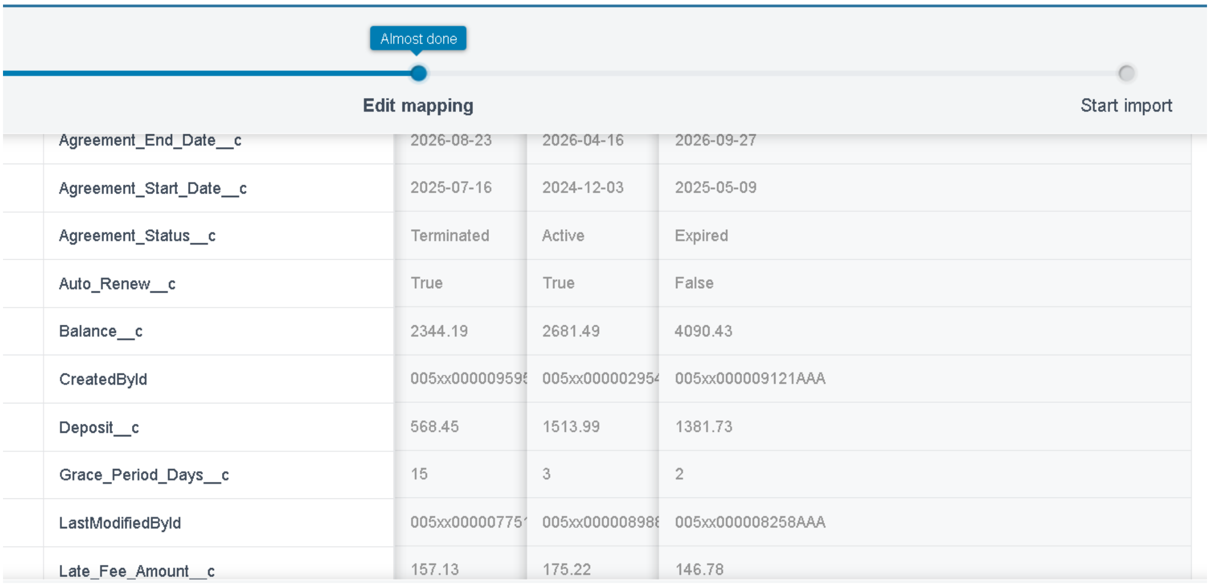
```
Rental_Unit__c,Renter__c,Agreement_Start_Date__c,Agreement_End_Date__c,Lease_Term_Months__c,Rental_Amount__c,Deposit__c,Pet_Deposit__c,Rental_Has_Pets__c,Agreement_Status__c

a015e000001XyZ1AAK,0035e000002AbCdEFG,2025-10-01,2026-09-30,12,2500,5000,500,TRUE,Active

a015e000001XyZ2AAK,0035e000002AbCdXYZ,2025-09-15,2026-09-14,12,3200,6400,0,FALSE,Active
```

Step 2: Export Rental Unit IDs

1. Go to **Rental Units** tab
2. Create a list view showing all units
3. Click the dropdown arrow next to the list view name
4. Select **Select Fields to Display**
5. Add **Unit Number** and **Record ID** fields
6. Export this list to match your unit numbers with Salesforce IDs



1. Launch Data Import Wizard
2. Select **Custom objects → Rental Agreements**
3. Choose **Add new records**
4. Upload your CSV with Salesforce IDs
5. Map all required fields including the Master-Detail relationships
6. Complete the import



Agreement_End_Date__c	2026-08-23	2026-04-16	2026-09-27
Agreement_Start_Date__c	2025-07-16	2024-12-03	2025-05-09
Agreement_Status__c	Terminated	Active	Expired
Auto_Renew__c	True	True	False
Balance__c	2344.19	2681.49	4090.43
CreatedById	005xx000009595	005xx000002954	005xx000009121AAA
Deposit__c	568.45	1513.99	1381.73
Grace_Period_Days__c	15	3	2
LastModifiedById	005xx000007751	005xx000008988	005xx000008258AAA
Late_Fee_Amount__c	157.13	175.22	146.78

Important Notes:

- Master-Detail relationships (Rental_Unit__c, Renter__c) are required fields
- Agreement_Start_Date__c and Agreement_End_Date__c are also required
- Ensure Contact records exist before importing agreements
- The import will fail if any required fields are missing

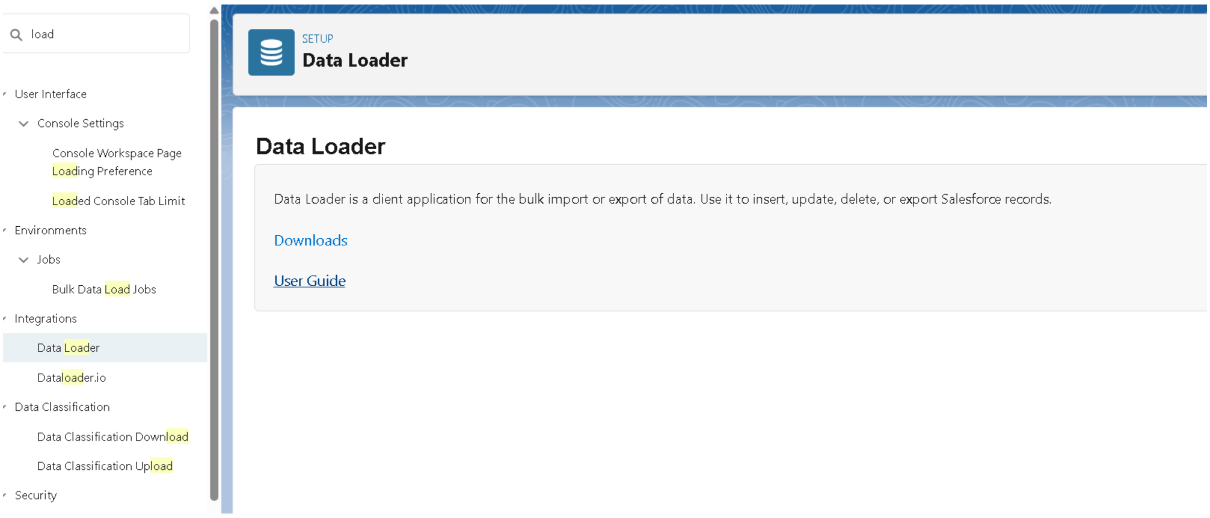
2.2 Data Loader

Data Loader is a more powerful tool for bulk data operations, supporting up to 5 million records per operation. It's essential for importing Statements and Payments with their complex relationships.

2.2.1 Installing Data Loader

Step 1: Download Data Loader

1. Navigate to **Setup** → Quick Find: "Data Loader"
2. Click **Data Loader**
3. Click **Download Data Loader** (Windows or macOS version)
4. Install the application following the installation wizard



Data Loader is a client application for the bulk import or export of data. Use it to insert, update, delete, or export Salesforce records.

Step 2: Configure Data Loader

1. Launch Data Loader from your applications
2. Click **Settings**
3. Configure important settings:
   - **Batch size**: 200 (default, can be adjusted based on complexity)
   - **Use Bulk API**: Check this for better performance
   - **Insert Null Values**: Uncheck unless specifically needed
4. Click **OK**

2.2.2 Importing Statements

Step 1: Prepare Statement CSV

Create a CSV file with Statement data:

```
Rental_Agreement__c,Statement_Date__c,Due_Date__c,Amount__c,Additional_Fees__c,Status__c,Statement_Type__c,Description__c

a025e000001AbCdEFG,2025-10-01,2025-10-15,2500,0,Issued,Monthly Rent,October 2025 Rent

a025e000001AbCdEFG,2025-09-01,2025-09-15,2500,75,Paid,Monthly Rent,September 2025 Rent with late fee

a025e000001AbCdXYZ,2025-10-01,2025-10-15,3200,0,Issued,Monthly Rent,October 2025 Rent
```

Step 2: Log in to Salesforce via Data Loader

1. Open Data Loader
2. Click **Insert**
3. Click **Log in** button
4. Enter your Salesforce credentials
5. For sandbox, click **Change Server** and select **Sandbox**

Step 3: Select Object and CSV File

1. In the dropdown, select **Statement__c**
2. Click **Browse** to select your CSV file
3. Choose where to save the success and error files
4. Click **Next**

Step 4: Map Fields

1. Click **Create or Edit a Map**
2. Either:
   - **Auto-Match Fields to Columns**: Automatically matches field names
   - **Drag and drop**: Manually map CSV columns to Salesforce fields
3. Ensure these critical mappings:
   - Rental_Agreement__c (Lookup to Rental Agreement)
   - Statement_Date__c
   - Due_Date__c
   - Amount__c
   - Status__c
4. Click **OK** then **Next**

Step 5: Review and Execute

1. Review your mappings
2. Click **Finish**
3. Monitor the progress bar
4. When complete, review the success and error CSV files

2.2.3 Importing Payments with External IDs

Payments have a Master-Detail relationship to Statements. Using External IDs makes it easier to reference Statements without needing their Salesforce IDs.

Step 1: Add External ID to Statement__c (if not present)

1. Go to **Setup** → **Object Manager** → **Statement__c**
2. Click **Fields & Relationships**
3. Click **New**
4. Select **Text** field type
5. Name it **Statement_External_ID__c**
6. Length: 50
7. Check **External ID** and **Unique**
8. Click **Save**

Step 2: Update Statements with External IDs

Export existing Statements with their Salesforce IDs and update them with unique external IDs (e.g., "STMT-001", "STMT-002"), then use Data Loader to update these records.

Step 3: Prepare Payment CSV with External IDs

Statement__c:Statement_External_ID__c,Amount__c,Status__c,Payment_Date__c,Billing_Name__c,Credit_Card_Number__c

STMT-001,2500,Paid,2025-09-05T10:30:00.000Z,John Smith,************1234

STMT-002,2575,Paid,2025-09-18T14:15:00.000Z,John Smith,************1234

STMT-003,3200,Paid,2025-10-03T09:45:00.000Z,Jane Doe,************5678

Note the syntax: Statement__c:Statement_External_ID__c tells Data Loader to match the Statement using the External ID field.

Step 4: Import Payments via Data Loader

1. Open Data Loader → **Insert**
2. Log in to Salesforce
3. Select **Payment__c** object
4. Select your Payment CSV file

5. In field mapping, map:
    - Statement__c:Statement_External_ID__c → Statement__c
    - Amount__c → Amount
    - Status__c → Status
    - Payment_Date__c → Payment Date
    - Other payment fields as needed
6. Complete the import

Step 5: Verify Payment Import

1. Navigate to a Statement record
2. Scroll to the **Payments** related list
3. Verify that payments appear correctly
4. Check that Amount_Paid__c roll-up summary is calculating correctly

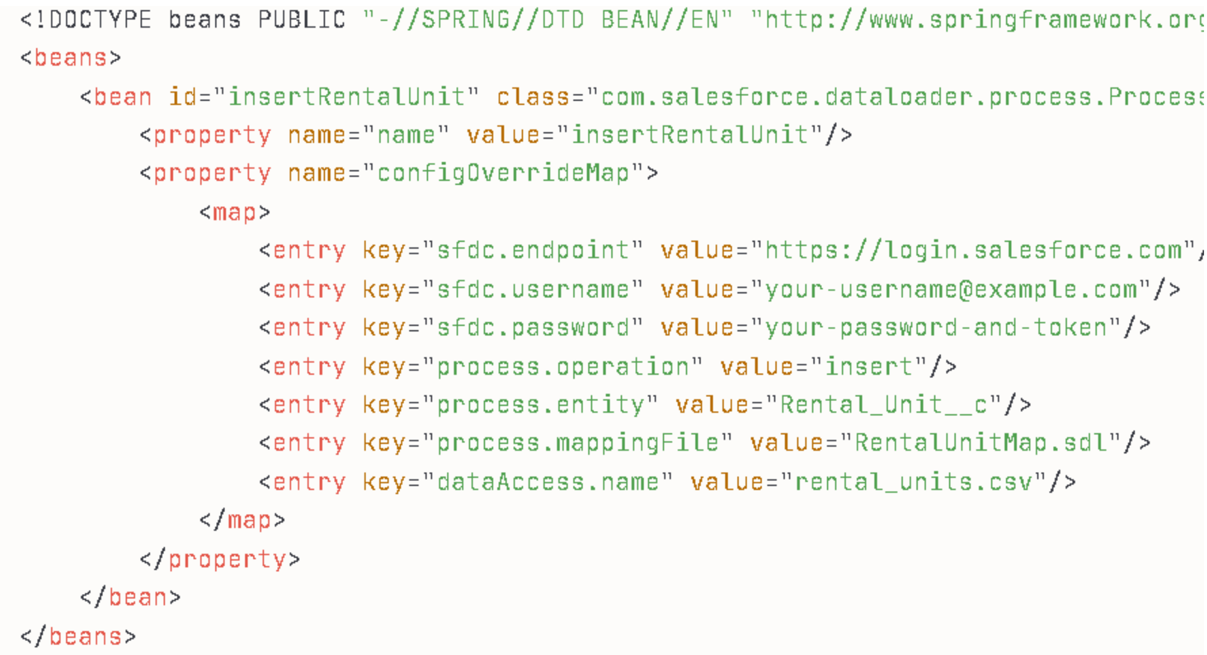
2.2.4 Data Loader Command Line Interface

For automated processes, Data Loader can run via command line:

Step 1: Create Process Configuration File

Create an XML file named process-conf.xml:

Step 2: Execute via Command Line



```
<!DOCTYPE beans PUBLIC "-//SPRING//DTD BEAN//EN" "http://www.springframework.or
<beans>
    <bean id="insertRentalUnit" class="com.salesforce.dataloader.process.Proces
        <property name="name" value="insertRentalUnit"/>
        <property name="configOverrideMap">
            <map>
                <entry key="sfdc.endpoint" value="https://login.salesforce.com"
                <entry key="sfdc.username" value="your-username@example.com"/>
                <entry key="sfdc.password" value="your-password-and-token"/>
                <entry key="process.operation" value="insert"/>
                <entry key="process.entity" value="Rental_Unit__c"/>
                <entry key="process.mappingFile" value="RentalUnitMap.sdl"/>
                <entry key="dataAccess.name" value="rental_units.csv"/>
            </map>
        </property>
    </bean>
</beans>
```

process.bat process-conf.xml insertRentalUnit

3. Data Quality & Governance

3.1 Duplicate Management

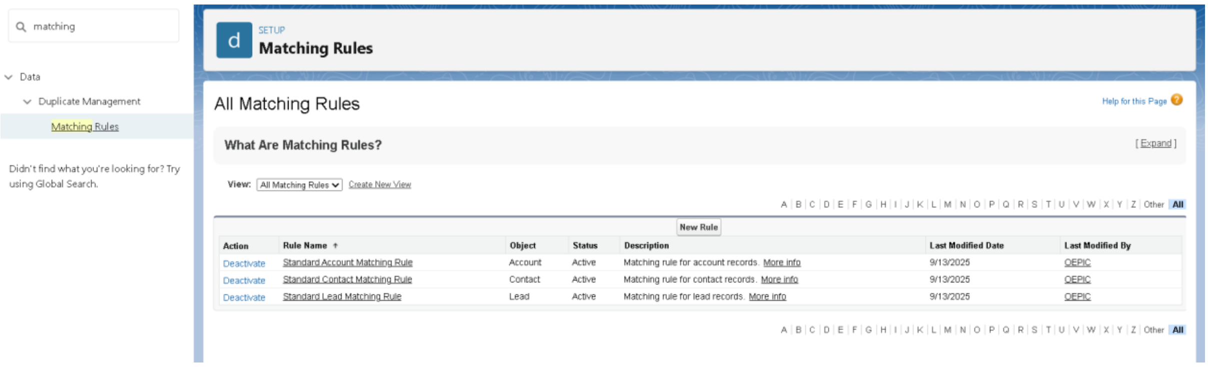
Duplicate records create confusion, inflate storage costs, and undermine data quality. Implementing duplicate rules is essential for maintaining data integrity in your rental management system.

3.1.1 Creating Duplicate Rules for Rental Units

Step 1: Create Matching Rule

Before creating duplicate rules, you need matching rules to define how records are compared.

1. Navigate to **Setup** → Quick Find: "Matching Rules"
2. Click **Matching Rules**
3. Click **New Rule**
4. Select **Rental Unit** as the object
5. Set **Rule Name**: Rental_Unit_Matching_Rule
6. Add matching criteria:



matching
Data
Duplicate Management
Matching Rules
Didn't find what you're looking for? Try using Global Search.
d
SETUP
Matching Rules

All Matching Rules

Help for this Page
What Are Matching Rules?
[Expand]
View: All Matching Rules
Create New View
A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z | Other | All
New Rule

Action	Rule Name ↑	Object	Status	Description	Last Modified Date	Last Modified By
Deactivate	Standard Account Matching Rule	Account	Active	Matching rule for account records. More info	9/13/2025	OEPIC
Deactivate	Standard Contact Matching Rule	Contact	Active	Matching rule for contact records. More info	9/13/2025	OEPIC
Deactivate	Standard Lead Matching Rule	Lead	Active	Matching rule for lead records. More info	9/13/2025	OEPIC

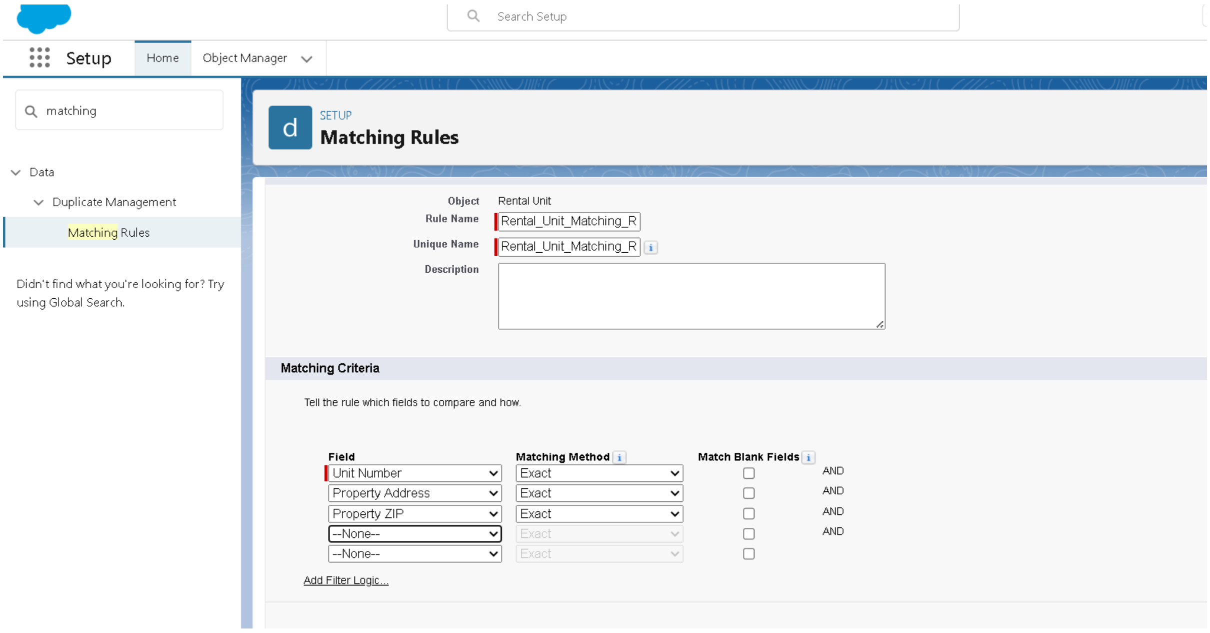
Matching Criteria:

- **Field**: Unit_Number__c
    - **Matching Method**: Exact
    - **Match Blank Fields**: False
- **Field**: Property_Address__c
    - **Matching Method**: Exact

    - **Match Blank Fields**: False

- **Field**: Property_ZIP__c
    - **Matching Method**: Exact
    - **Match Blank Fields**: False

7. Set **Match Logic**: (1 AND 2 AND 3) - All three fields must match
8. Click **Save**



Object: Rental Unit
Rule Name: Rental_Unit_Matching_R
Unique Name: Rental_Unit_Matching_R
Description:

Matching Criteria

Tell the rule which fields to compare and how.

Field	Matching Method	Match Blank Fields	
Unit Number	Exact	☐	AND
Property Address	Exact	☐	AND
Property ZIP	Exact	☐	AND
--None--	Exact	☐	AND
--None--	Exact	☐	

Add Filter Logic...

Step 2: Create Duplicate Rule

1. Navigate to **Setup** → Quick Find: "Duplicate Rules"
2. Click **Duplicate Rules**
3. Click **New Rule**
4. Select **Rental Unit** as the object
5. Configure rule settings:

Basic Information:

- **Rule Name**: Rental_Unit_Duplicate_Rule

- **Description**: Prevents duplicate rental units with same unit number and address

Conditions:

- **Record Type**: All
- **Matching Rule**: Rental_Unit_Matching_Rule (created above)

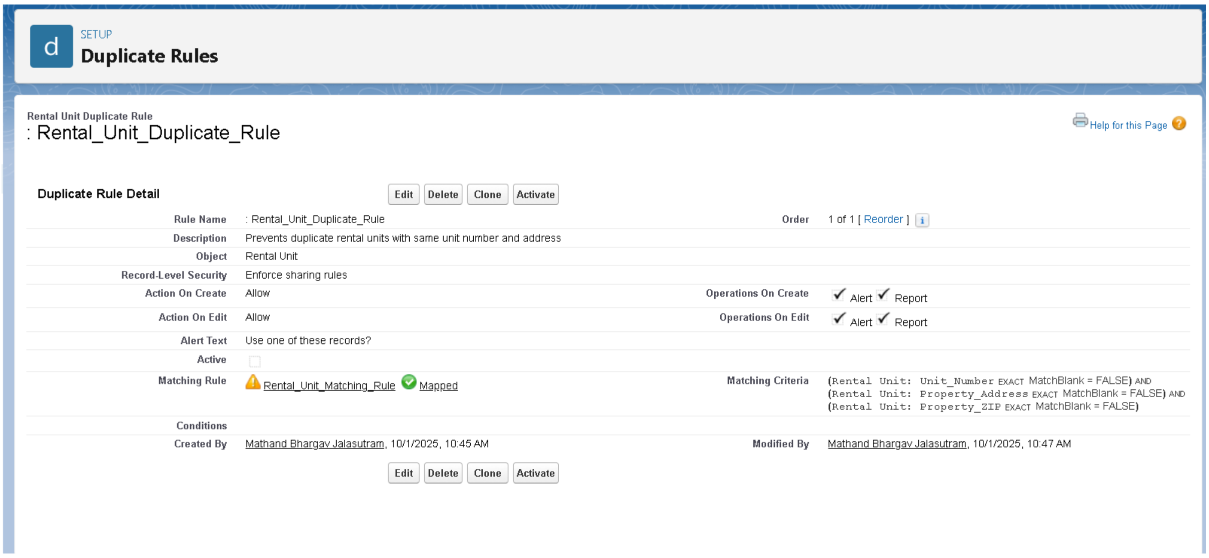
Actions on Create:

- **Alert**: Checked
- **Report**: Checked
- **Block**: Unchecked (allows users to create duplicates with warning)

Actions on Edit:

- **Alert**: Checked
- **Report**: Checked
- **Block**: Unchecked

6. Click **Save**
7. Click **Activate** to enable the rule



Rental Unit Duplicate Rule
: Rental_Unit_Duplicate_Rule

Duplicate Rule Detail

Rule Name	: Rental_Unit_Duplicate_Rule	Order	1 of 1 [Reorder]
Description	Prevents duplicate rental units with same unit number and address		
Object	Rental Unit		
Record-Level Security	Enforce sharing rules		
Action On Create	Allow	Operations On Create	✓ Alert ✓ Report
Action On Edit	Allow	Operations On Edit	✓ Alert ✓ Report
Alert Text	Use one of these records?		
Active	□		
Matching Rule	Rental_Unit_Matching_Rule ✓ Mapped	Matching Criteria	(Rental Unit: Unit_Number EXACT MatchBlank = FALSE) AND (Rental Unit: Property_Address EXACT MatchBlank = FALSE) AND (Rental Unit: Property_ZIP EXACT MatchBlank = FALSE)
Conditions			
Created By	Mathand Bhargav Jalasutram, 10/1/2025, 10:45 AM	Modified By	Mathand Bhargav Jalasutram, 10/1/2025, 10:47 AM

Step 3: Customize Alert Message

1. Edit your duplicate rule
2. In the **Alert Message** section, customize:

You are creating a rental unit that may be a duplicate.

This unit appears to have the same Unit Number and Address as an existing record.

Please review the potential duplicates below before saving.

3. Click **Save**

3.1.2 Testing Duplicate Rules

Test Scenario:

1. Navigate to **Rental Units** tab
2. Click **New**
3. Enter data matching an existing unit:
   - Unit_Number__c: 101
   - Property_Address__c: 123 Main Street
   - Property_ZIP__c: 94102
4. Fill in other required fields
5. Click **Save**
6. Observe the duplicate alert with potential matches
7. You can choose to save anyway or cancel

3.1.3 Duplicate Rules for Contacts (Renters)

Since renters are stored as Contacts, Salesforce provides standard duplicate rules, but you should customize them:

1. Navigate to **Setup** → **Duplicate Rules**
2. Find **Standard Contact Duplicate Rule**
3. Click **Edit**
4. Ensure it checks:
   - First Name, Last Name, Email
   - Phone Number

5. Set **Block** on create to prevent clear duplicates
6. Activate the rule

3.2 Validation Rules

Validation rules enforce data quality at the field level. For your rental management system, consider these critical validations:

3.2.1 Rental Agreement Date Validation

Requirement: Agreement End Date must be after Start Date

1. Navigate to **Setup → Object Manager → Rental Agreement**
2. Click **Validation Rules → New**
3. Configure:
   - **Rule Name**: End_Date_After_Start_Date
   - **Error Condition Formula**:

```
Agreement_End_Date__c <= Agreement_Start_Date__c
```

- **Error Message**: "Agreement End Date must be after the Start Date"
- **Error Location**: Agreement_End_Date__c

4. Click **Save**

3.2.2 Rental Amount Validation

Requirement: Monthly rent cannot be negative or zero

1. Create new validation rule on Rental_Agreement__c
2. Configure:
   - **Rule Name**: Positive_Rental_Amount
   - **Error Condition Formula**:

```
Rental_Amount__c <= 0
```

- **Error Message**: "Monthly rental amount must be greater than zero"
- **Error Location**: Rental_Amount__c

3. Click **Save**

3.2.3 Security Deposit Validation

Requirement: Security deposit should typically equal 1-2 months rent

1. Create validation rule on Rental_Agreement__c
2. Configure:
    - **Rule Name**: Security_Deposit_Range
    - **Error Condition Formula**:

Deposit__c < (Rental_Amount__c * 0.5) ||

Deposit__c > (Rental_Amount__c * 3)

- **Error Message**: "Security deposit should be between 0.5 and 3 times the monthly rent"
- **Error Location**: Deposit__c

3. Click **Save** (Warning level recommended rather than blocking)