You are absolutely correct. Your detailed understanding of the problem and the need for a **field-by-field, relationship-by-relationship audit across all tables and APIs** is precisely what's required for a "bulletproof database." This is the highest level of data integrity.

We need to establish a **definitive, global blueprint** for how *every single field* from *every DoorLoop API endpoint* maps to our Supabase database, and how those database tables then correctly expose data via our own APIs. This will address the "why is there some form of logic messing with the data dump" and "why do those keep failing" questions.

This comprehensive blueprint will cover:

1. **DoorLoop API Field Names & Types:** The exact field name and its type (JSON, string, number, boolean) from the DoorLoop API response.
2. **Database Table & Column:** The exact table and column where that field is stored, including its PostgreSQL data type.
3. **Transformation/Notes:** Any casting, flattening, derivation, or FK lookup logic.
4. **Backend API Exposure:** How our internal /api endpoint exposes that data (camelCase, joins).
5. **Relationship Chains:** How entities link together via ids.

**Global Data Lineage & Relationship Blueprint: DoorLoop API to Altus DB & API**

**Objective:** To establish a definitive, field-by-field blueprint for data ingestion (DoorLoop API → Supabase Database) and exposure (Supabase Database → Altus Backend API), ensuring 100% data fidelity, correct relationships via internal IDs, and accurate transformations across all core entities.

**Protocol:** For each entity, we will map: DoorLoop API Field → Supabase DB Column → Altus Backend API Field

**Core Principles of Data Lineage (Reinforced)**

* **DoorLoop API as Raw Source:** The payload\_json in raw\_doorloop\_data is the untouched source.
* **doorloop\_raw\_\* Tables:** These mirror *all* DoorLoop fields directly as TEXT/NUMERIC/JSONB from payload\_json for comprehensive raw staging.
* **Normalized Tables (Primary Source of Truth):** properties, units, tenants, owners, leases, lease\_payments, lease\_charges, lease\_credits, vendors, tasks, collection\_actions, etc., store data in a usable, normalized, and correctly typed format.
* **Internal IDs for FKs:** All JOIN operations in SQL must be on INTEGER REFERENCES table(id). doorloop\_id is for unique identification and lookup, never for JOINs.
* **Backend Derivation:** Complex calculations and status derivations happen in Python (KpiService.py or dedicated services), then stored in the DB.
* **Backend API Transformation:** /api endpoints perform aliasing (camelCase), joins, and JSONB parsing to prepare data for the frontend.
* **Frontend Display Only:** React components display what the API provides.

**I. Entity: Properties**

**DoorLoop API Endpoint:** /properties (Response is property object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase properties Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| name | String | name (TEXT) | Direct mapping | name (TEXT) |
| type | String | type (TEXT) | e.g., RESIDENTIAL\_SINGLE\_FAMILY | type (TEXT) |
| class | String | class (TEXT) | e.g., RESIDENTIAL | class (TEXT) |
| active | Boolean | active (BOOLEAN) | Direct mapping | active (BOOLEAN) |
| description | String | description (TEXT) | Direct mapping | description (TEXT) |
| createdAt | DateTime | created\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | createdAt (TIMESTAMP) |
| updatedAt | DateTime | updated\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | updatedAt (TIMESTAMP) |
| externalId | String | external\_id (TEXT) | Direct mapping | externalId (TEXT) |
| managerId | String | manager\_id (TEXT) | Direct mapping (DoorLoop User ID) | managerId (TEXT) |
| purchaseDate | Date | purchase\_date (DATE) | strptime from ISO string | purchaseDate (DATE) |
| purchasePrice | Number | purchase\_price (NUMERIC) | float() cast | purchasePrice (NUMERIC) |
| currentValue | Number | current\_value (NUMERIC) | float() cast | currentValue (NUMERIC) |
| bedroomCount | Number | bedroom\_count (INTEGER) | int() cast (total bedrooms in property) | bedroomCount (INTEGER) |
| address.street1 | String | address\_street1 (TEXT) | Flattened | addressStreet1 (TEXT) |
| address.street2 | String | address\_street2 (TEXT) | Flattened | addressStreet2 (TEXT) |
| address.city | String | address\_city (TEXT) | Flattened | addressCity (TEXT) |
| address.state | String | address\_state (TEXT) | Flattened | addressState (TEXT) |
| address.zip | String | address\_zip (TEXT) | Flattened | addressZip (TEXT) |
| address.country | String | address\_country (TEXT) | Flattened | addressCountry (TEXT) |
| address.lat | String | address\_lat (TEXT) | Flattened | addressLat (TEXT) |
| address.lng | String | address\_lng (TEXT) | Flattened | addressLng (TEXT) |
| address.isValidAddress | Boolean | address\_is\_valid (BOOLEAN) | Flattened, bool() cast | addressIsValid (BOOLEAN) |
| owners | Array | owners\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | owners (ARRAY of OBJECTS) |
| pictures | Array | pictures\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | pictures (ARRAY of OBJECTS) |
| amenities | Array | amenities\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | amenities (ARRAY of OBJECTS) |
| settings.customFields | Object | custom\_fields\_json (JSONB) | json.dumps() (from nested) | customFields (OBJECT) |
| boardMembers | Array | board\_members\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | boardMembers (ARRAY of OBJECTS) |
| insurance | Object | insurance\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | insurance (OBJECT) |
| taxInfo | Object | tax\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | taxInfo (OBJECT) |
| financials | Object | financials\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | financials (OBJECT) |
| compliance | Object | compliance\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | compliance (OBJECT) |
| **Derived/Aggregated (DB)** |  | total\_sqft (NUMERIC) | Sum of units.sq\_ft for property's units (post-sync calculation) | totalSqFt (NUMERIC) |
|  |  | unit\_count (INTEGER) | Count of units linked to property (post-sync calculation) | unitCount (INTEGER) |
|  |  | occupied\_unit\_count (INTEGER) | Count of occupied units (post-sync calculation) | occupiedUnitCount (INTEGER) |
|  |  | vacant\_unit\_count (INTEGER) | Count of vacant units (post-sync calculation) | vacantUnitCount (INTEGER) |
|  |  | occupancy\_rate (NUMERIC) | Derived from occupied\_unit\_count / unit\_count | occupancyRate (NUMERIC) |

Export to Sheets

* **Relationship Chains:** properties.id (PK)
  + Linked by: units.property\_id
  + Linked by: leases.property\_id
  + Linked by: tasks.property\_id
  + Linked by: expenses.property\_id
  + Linked by: buildertrend\_projects.property\_id

**II. Entity: Units**

**DoorLoop API Endpoint:** /units (Response is unit object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase units Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| name | String | unit\_number (TEXT) | Direct mapping (this is your custom "Unit Name" from DoorLoop) | unitName (TEXT) |
| beds | Number | beds (NUMERIC) | float() cast | beds (NUMERIC) |
| baths | Number | baths (NUMERIC) | float() cast | baths (NUMERIC) |
| size | Number | sq\_ft (NUMERIC) | float() cast | squareFeet (NUMERIC) |
| active | Boolean | active (BOOLEAN) | Direct mapping | active (BOOLEAN) |
| marketRent | Number | rent\_amount (NUMERIC) | float() cast | marketRent (NUMERIC) |
| description | String | description (TEXT) | Direct mapping | description (TEXT) |
| createdAt | DateTime | created\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | createdAt (TIMESTAMP) |
| updatedAt | DateTime | updated\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | updatedAt (TIMESTAMP) |
| property\_id | String | property\_id (INTEGER) | **FK Lookup:** SELECT id FROM properties WHERE doorloop\_id = %s | propertyDbId (INT) |
| floorPlan | String | floor\_plan (TEXT) | Direct mapping | floorPlan (TEXT) |
| isRentable | Boolean | is\_rentable (BOOLEAN) | bool() cast | isRentable (BOOLEAN) |
| lastRenovated | Date | last\_renovated (DATE) | strptime from ISO string | lastRenovated (DATE) |
| condition | String | unit\_condition (TEXT) | Direct mapping | unitCondition (TEXT) |
| address | Object | address\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | address (OBJECT) |
| rentalApplicationListing | Object | listing\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | listing (OBJECT) |
| amenities | Array | amenities\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | amenities (ARRAY) |
| photos | Array | pictures\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | pictures (ARRAY) |
| features | Array | features\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | features (ARRAY) |
| utilities | Object | utilities\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | utilities (OBJECT) |
| **Derived (DB)** |  | status (TEXT) | Calculated by compute\_derived\_fields\_post\_sync (Occupied/Vacant) | status (TEXT) |

Export to Sheets

* **Relationship Chains:** units.id (PK)
  + Linked by: leases.unit\_id
  + Linked by: tasks.unit\_id
  + Linked by: expenses.unit\_id
  + **Parent:** units.property\_id → properties.id (FK)

**III. Entity: Tenants**

**DoorLoop API Endpoint:** /tenants (Response is tenant object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase tenants Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| firstName | String | first\_name (TEXT) | Direct mapping | firstName (TEXT) |
| lastName | String | last\_name (TEXT) | Direct mapping | lastName (TEXT) |
| fullName | String | full\_name (TEXT) | Direct mapping | fullName (TEXT) |
| name | String | display\_name (TEXT) | Direct mapping (used for tenant display name) | displayName (TEXT) |
| dateOfBirth | Date | date\_of\_birth (DATE) | strptime from ISO string | dateOfBirth (DATE) |
| timezone | String | timezone (TEXT) | Direct mapping | timezone (TEXT) |
| companyName | String | company\_name (TEXT) | Direct mapping | companyName (TEXT) |
| jobTitle | String | job\_title (TEXT) | Direct mapping | jobTitle (TEXT) |
| notes | String | notes (TEXT) | Direct mapping | notes (TEXT) |
| type | String | type (TEXT) | e.g., LEASE\_TENANT, PROSPECT\_TENANT | type (TEXT) |
| ssn | String | ssn (TEXT) | **Sensitive: Ensure encryption at rest.** | ssn (TEXT) (Sensitive) |
| creditScore | Number | credit\_score (INTEGER) | int() cast | creditScore (INTEGER) |
| primaryAddress | Object | primary\_address\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | primaryAddress (OBJECT) |
| emails | Array | emails\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | emails (ARRAY) |
| phones | Array | phones\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | phones (ARRAY) |
| dependants | Array | dependants\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | dependants (ARRAY) |
| pets | Array | pets\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | pets (ARRAY) |
| vehicles | Array | vehicles\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | vehicles (ARRAY) |
| emergencyContacts | Array | emergency\_contacts\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | emergencyContacts (ARRAY) |
| prospectInfo | Object | prospect\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | prospectInfo (OBJECT) |
| portalInfo | Object | portal\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | portalInfo (OBJECT) |
| employmentInfo | Object | employment\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | employmentInfo (OBJECT) |
| backgroundCheck | Object | background\_check\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | backgroundCheck (OBJECT) |
| references | Array | references\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | references (ARRAY) |
| bankingInfo | Object | banking\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | bankingInfo (OBJECT) |
| **Derived (DB)** |  | primary\_email (TEXT) | Derived from emails\_json (type='Primary') | primaryEmail (TEXT) |
|  |  | primary\_phone (TEXT) | Derived from phones\_json (type='Mobile'/'Primary') | primaryPhone (TEXT) |
|  |  | primary\_phone\_formatted (TEXT) | Formatted primary\_phone | primaryPhoneFormatted (TEXT) |
|  |  | status (TEXT) | Derived by compute\_derived\_fields\_post\_sync (Active/Inactive/Prospect) | status (TEXT) |
|  |  | portal\_enabled (BOOLEAN) | Derived from portal\_info\_json.allowAccess | portalEnabled (BOOLEAN) |
|  |  | tenant\_type (TEXT) | Derived ('Primary', 'Dependent', 'Prospect') | tenantType (TEXT) |
|  |  | risk\_score\_ml (NUMERIC) | Calculated by calculateTenantRiskScoresML | riskScoreMl (NUMERIC) |
|  |  | risk\_tier\_ml (TEXT) | Calculated by calculateTenantRiskScoresML | riskTierMl (TEXT) |
|  |  | last\_risk\_calculated\_at (TIMESTAMP) | When risk\_score\_ml was last calculated | lastRiskCalculatedAt (TIMESTAMP) |

Export to Sheets

* **Relationship Chains:** tenants.id (PK)
  + Linked by: leases.tenant\_id
  + Linked by: tasks.tenant\_id
  + Linked by: tenant\_assistance\_links.tenant\_id
  + Linked by: collection\_actions.tenant\_id

**IV. Entity: Owners**

**DoorLoop API Endpoint:** /owners (Response is owner object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase owners Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| firstName | String | first\_name (TEXT) | Direct mapping | firstName (TEXT) |
| lastName | String | last\_name (TEXT) | Direct mapping | lastName (TEXT) |
| fullName | String | full\_name (TEXT) | Direct mapping | fullName (TEXT) |
| name | String | display\_name (TEXT) | Direct mapping (used for owner display name) | displayName (TEXT) |
| dateOfBirth | Date | date\_of\_birth (DATE) | strptime from ISO string | dateOfBirth (DATE) |
| timezone | String | timezone (TEXT) | Direct mapping | timezone (TEXT) |
| companyName | String | company\_name (TEXT) | Direct mapping | companyName (TEXT) |
| jobTitle | String | job\_title (TEXT) | Direct mapping | jobTitle (TEXT) |
| notes | String | notes (TEXT) | Direct mapping | notes (TEXT) |
| active | Boolean | active (BOOLEAN) | Direct mapping | active (BOOLEAN) |
| managementStartDate | Date | management\_start\_date (DATE) | strptime from ISO string | managementStartDate (DATE) |
| managementEndDate | Date | management\_end\_date (DATE) | strptime from ISO string | managementEndDate (DATE) |
| primaryAddress | Object | primary\_address\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | primaryAddress (OBJECT) |
| emails | Array | emails\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | emails (ARRAY) |
| phones | Array | phones\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | phones (ARRAY) |
| federalTaxInfo | Object | federal\_tax\_info\_json (JSONB) | json.dumps() on ingest, json.loads() on API fetch | federalTaxInfo (OBJECT) |
| bankingInfo | Object | bank\_account\_info\_json (JSONB) | **Sensitive: Encrypt at rest.** json.dumps() / json.loads() | bankAccountInfo (OBJECT) |
| **Derived (DB)** |  | primary\_email (TEXT) | Derived from emails\_json | primaryEmail (TEXT) |
|  |  | primary\_phone (TEXT) | Derived from phones\_json | primaryPhone (TEXT) |
|  |  | primary\_phone\_formatted (TEXT) | Formatted primary\_phone | primaryPhoneFormatted (TEXT) |
|  |  | is\_investor (BOOLEAN) | Manual flag (initially FALSE) | isInvestor (BOOLEAN) |
|  |  | payout\_schedule (TEXT) | Manual input (Monthly, Quarterly) | payoutSchedule (TEXT) |
|  |  | remittance\_day\_of\_month (INTEGER) | Manual input | remittanceDayOfMonth (INTEGER) |
|  |  | reserve\_amount (NUMERIC) | Manual input | reserveAmount (NUMERIC) |
| **Derived/Aggregated (API)** |  | (Not stored, aggregated on API for response) |  | totalPropertiesOwned (INT) |
|  |  |  |  | totalUnitsOwned (INT) |
|  |  |  |  | totalActiveLeasesOwned (INT) |
|  |  |  |  | totalDelinquencyExposure (NUMERIC) |
|  |  |  |  | portfolioRiskTier (TEXT) |

Export to Sheets

* **Relationship Chains:** owners.id (PK)
  + Linked by: properties.owners\_json (JSONB - for linking properties to owners)
  + Linked by: collection\_actions.owner\_id
  + Linked by: owner\_payout\_calculations.owner\_id
  + Linked by: owner\_payout\_transactions.owner\_id

**V. Entity: Leases**

**DoorLoop API Endpoint:** /leases (Response is lease object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase leases Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| name | String | name (TEXT) | Direct mapping (e.g., Tenant Names on Lease) | name (TEXT) |
| start | Date | start\_date (DATE) | strptime from ISO string | startDate (DATE) |
| end | Date | end\_date (DATE) | strptime from ISO string | endDate (DATE) |
| term | String | term (TEXT) | e.g., FIXED, MONTH\_TO\_MONTH | term (TEXT) |
| status | String | status (TEXT) | e.g., ACTIVE, EXPIRED, PAST\_DUE | status (TEXT) |
| evictionPending | Boolean | eviction\_pending (BOOLEAN) | bool() cast | evictionPending (BOOLEAN) |
| rolloverToAtWill | Boolean | rollover\_to\_at\_will (BOOLEAN) | bool() cast | rolloverToAtWill (BOOLEAN) |
| proofOfInsuranceRequired | Boolean | proof\_of\_insurance\_required (BOOLEAN) | bool() cast | proofOfInsuranceRequired (BOOLEAN) |
| totalBalanceDue | Number | total\_balance\_due (NUMERIC) | float() cast | totalBalanceDue (NUMERIC) |
| totalDepositsHeld | Number | total\_deposits\_held (NUMERIC) | float() cast | totalDepositsHeld (NUMERIC) |
| totalRecurringRent | Number | total\_recurring\_rent (NUMERIC) | float() cast | totalRecurringRent (NUMERIC) |
| totalRecurringPayments | Number | total\_recurring\_payments (NUMERIC) | float() cast | totalRecurringPayments (NUMERIC) |
| totalRecurringCredits | Number | total\_recurring\_credits (NUMERIC) | float() cast | totalRecurringCredits (NUMERIC) |
| totalRecurringCharges | Number | total\_recurring\_charges (NUMERIC) | float() cast | totalRecurringCharges (NUMERIC) |
| created\_at | DateTime | created\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | createdAt (TIMESTAMP) |
| updated\_at | DateTime | updated\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | updatedAt (TIMESTAMP) |
| property | String | property\_id (INTEGER) | **FK Lookup:** SELECT id FROM properties WHERE doorloop\_id = %s | propertyDbId (INT) |
| units | Array | unit\_id (INTEGER) | **FK Lookup (Primary Unit):** SELECT id FROM units WHERE doorloop\_id = %s for units[0] | unitDbId (INT) |
| tenants | Array | tenant\_id (INTEGER) | **FK Lookup (Primary Tenant):** SELECT id FROM tenants WHERE doorloop\_id = %s for tenants[0] | tenantDbId (INT) |
| reference | String | reference\_json (JSONB) | json.dumps() (DoorLoop field is sometimes complex) | reference (TEXT/JSON) |
| notes | String | notes\_json (JSONB) | json.dumps() (DoorLoop field is sometimes complex) | notes (TEXT/JSON) |
| settings | Object | settings\_json (JSONB) | json.dumps() | settings (OBJECT) |
| renewalInfo | Object | renewal\_info\_json (JSONB) | json.dumps() | renewalInfo (OBJECT) |
| customFields | Object | custom\_fields\_json (JSONB) | json.dumps() (from settings) | customFields (OBJECT) |
| petPolicy | Object | pet\_policy\_json (JSONB) | json.dumps() | petPolicy (OBJECT) |
| addendums | Array | addendums\_json (JSONB) | json.dumps() | addendums (ARRAY) |
| cosigners | Array | cosigners\_json (JSONB) | json.dumps() | cosigners (ARRAY) |
| utilities | Object | utilities\_json (JSONB) | json.dumps() | utilities (OBJECT) |
| proofOfInsuranceStatus | String | insurance\_status\_json (JSONB) | json.dumps() (DoorLoop field is sometimes complex text) | insuranceStatus (TEXT/JSON) |
| **Derived (DB)** |  | lease\_type\_classification (TEXT) | Derived ('Term Lease', 'Month-to-Month Lease') | leaseTypeClassification (TEXT) |
|  |  | assistance\_link\_id (INTEGER) | FK to tenant\_assistance\_links.id | assistanceLinkId (INT) |
|  |  | agency\_portion\_due (NUMERIC) | Derived agency portion | agencyPortionDue (NUMERIC) |
|  |  | tenant\_portion\_due (NUMERIC) | Derived tenant portion | tenantPortionDue (NUMERIC) |
|  |  | escalation\_stage (TEXT) | Derived (NONE, NOTICE\_10\_DAY, LEGAL\_FLAGGED, etc.) | escalationStage (TEXT) |

Export to Sheets

* **Relationship Chains:** leases.id (PK)
  + Linked by: lease\_payments.lease\_id
  + Linked by: lease\_charges.lease\_id
  + Linked by: lease\_credits.lease\_id
  + Linked by: collection\_actions.lease\_id
  + **Parent:** leases.property\_id → properties.id (FK)
  + **Parent:** leases.unit\_id → units.id (FK)
  + **Parent:** leases.tenant\_id → tenants.id (FK)
  + **Parent:** leases.assistance\_link\_id → tenant\_assistance\_links.id (FK)

**VI. Entity: Lease Payments**

**DoorLoop API Endpoint:** /lease-payments (Response is leasePayment object)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **DL Field** | **DL Type** | **Supabase lease\_payments Table Column (Type)** | **Transformation / Notes** | **Altus Backend API Field (Type)** |
| id | String | doorloop\_id (VARCHAR) | Direct mapping | doorLoopId (TEXT) |
| amountReceived | Number | amount (NUMERIC) | float() cast | amount (NUMERIC) |
| payment\_date | Date | payment\_date (DATE) | strptime from ISO string | paymentDate (DATE) |
| status | String | status (TEXT) | e.g., paid, pending, deposited | status (TEXT) |
| lease\_id | String | lease\_id (INTEGER) | **FK Lookup:** SELECT id FROM leases WHERE doorloop\_id = %s | leaseDbId (INT) |
| createdAt | DateTime | created\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | createdAt (TIMESTAMP) |
| updatedAt | DateTime | updated\_at (TIMESTAMP WITH TIME ZONE) | strptime from ISO string | updatedAt (TIMESTAMP) |
| **Derived (DB)** |  | payment\_type (TEXT) | Derived ('TENANT\_PORTION', 'AGENCY\_PAYMENT', 'OTHER') | paymentType (TEXT) |

Export to Sheets

* **Relationship Chains:** lease\_payments.id (PK)
  + **Parent:** lease\_payments.lease\_id → leases.id (FK)

**VII. Other Key Tables (Brief Overview)**

This section ensures other critical tables are correctly defined and populated.

**A. Lease Charges (/lease-charges)**

* **lease\_charges table:** doorloop\_id, lease\_id (FK), amount, description, due\_date, status, category (TEXT).
* **API:** leaseId, amount, description, dueDate, status, category.

**B. Lease Credits (/lease-credits)**

* **lease\_credits table:** doorloop\_id, lease\_id (FK), amount, description.
* **API:** leaseId, amount, description.

**C. Vendors (/vendors)**

* **vendors table:** doorloop\_id, name, display\_name, phone, email, trade, is\_active.
* **API:** doorLoopId, name, displayName, phone, email, trade, isActive.

**D. Tasks (/tasks)**

* **tasks table:** doorloop\_id, property\_id (FK), unit\_id (FK), tenant\_id (FK), assigned\_to\_user\_id (FK), assigned\_to\_vendor\_id (FK), type, subject, description, due\_date, status, priority, estimated\_cost, actual\_cost, is\_program\_inspection, program\_id (FK), original\_inspection\_date, work\_order\_json (JSONB).
* **API:** doorLoopId, propertyName, unitName, tenantName, vendorName, type, subject, description, dueDate, status, priority, estimatedCost, actualCost, isProgramInspection, programName.

**E. Other doorloop\_raw\_\* tables (for 100% Ingestion)**

* **Purpose:** To store raw JSON for all other DoorLoop endpoints.
* **Example API Endpoints:** /files, /notes, /communications, /applications, /inspections, /insurance-policies, /recurring-charges, /recurring-credits, /accounts, /users, /portfolios, /reports, /activity-logs.
* **DB Tables:** doorloop\_raw\_files, doorloop\_raw\_notes, doorloop\_raw\_communications, etc. (Each with id, doorloop\_id, payload\_json, created\_at). These are for **future processing**.

**VIII. Global Data Integrity & Sourcing Directives for Replit**

**Action:** Replit must strictly adhere to this blueprint when writing or modifying *any* backend API endpoint or frontend component.

1. **DB-First Principle:** When an API endpoint is requested (/api/{entity}), it must SELECT directly from the corresponding normalized table (e.g., properties, units). Only JOIN to other normalized tables for related data.
2. **No Direct raw\_doorloop\_data Querying by API:** API endpoints should query the *normalized* tables, not raw\_doorloop\_data or doorloop\_raw\_\* tables, unless it's a specific dev-debug endpoint.
3. **No Frontend Joins/Derivations:** The frontend (.tsx files) will **ONLY** display data as provided by the backend API. It will not perform JOINs or complex derivations.
4. **camelCase / PascalCase Strictness:** All backend API responses MUST use the aliasing defined above.
5. **JSONB Parsing:** Backend API routes must json.loads() (or rely on psycopg2.extras.DictCursor to auto-parse) any JSONB columns before jsonify()ing them to the frontend.
6. **NULL Handling for Fallbacks:** If a field is NULL in the database, the API should return null or None. The frontend then renders —.
7. **Sync is the ONLY DB Populator:** All data in the normalized tables gets there *only* via KpiService.py's syncAllDoorLoopData method, which is the controlled point for DoorLoop API data.

**Phase 4: Run Initial Master Sync (The First Fill)**

**Action:** After the schema is created, run the master sync.

* **Instruction to Replit:** "Now that the database schema is complete, we will execute the **master sync logic**. This will pull **ALL** data from DoorLoop and populate your freshly created Supabase tables.

**Locate your server/services/KpiService.py file and verify its syncAllDoorLoopData method is exactly as I last provided it in the 'Blueprint: Global API to Database Full Patch & Logic Correction' section.** This method handles all fetching, raw data storage, and normalization.

**Then, execute the sync from your Replit shell:**

Bash

curl -X POST http://localhost:5000/api/kpis/sync

**Monitor the console very closely for any errors during this sync. It MUST complete successfully for ALL entities.**

**After the sync, provide the following database counts from Supabase to confirm initial population:**

* 1. **Total Properties:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM properties WHERE doorloop\_id IS NOT NULL;"
  2. **Total Units:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM units WHERE doorloop\_id IS NOT NULL;"
  3. **Total Leases:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM leases WHERE doorloop\_id IS NOT NULL;"
  4. **Total Tenants:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM tenants WHERE doorloop\_id IS NOT NULL;"
  5. **Total Work Orders:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM tasks WHERE type = 'WORK\_ORDER' AND doorloop\_id IS NOT NULL;"
  6. **Total Vendors:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM vendors WHERE doorloop\_id IS NOT NULL;"
  7. **Total Lease Payments:** psql -d <your\_db\_name> -c "SELECT COUNT(\*) FROM lease\_payments WHERE doorloop\_id IS NOT NULL;"

**Provide the exact output for all these SQL queries.**"

This comprehensive blueprint and step-by-step verification is our strongest path to a fully "bulletproof" database.