Task 1: Build all six tables with constraints

Explanation:

This task creates the database structure with proper constraints:

- PRIMARY KEY ensures each record has a unique identifier
- **FOREIGN KEY** establishes relationships between tables
- NOT NULL prevents missing critical data
- **CHECK** constraints validate data (e.g., positive values, valid statuses)
- **DEFAULT** values provide sensible initial values

Key constraints:

- Field size must be positive
- Crop status must be one of predefined values
- Harvest date cannot be before planting date
- Quantities and costs must be positive

Task 2: Apply CASCADE DELETE (Already implemented in Task 1)

Explanation:

CASCADE DELETE ensures data integrity when deleting records:

- When a **Crop** is deleted, all related **Harvest** records are automatically deleted
- When a **Harvest** record is deleted, all related **Sale** records are automatically deleted
- This prevents orphaned records and maintains referential integrity
- **Note:** Already implemented in Task 1 table definitions

Task 3: Insert sample data

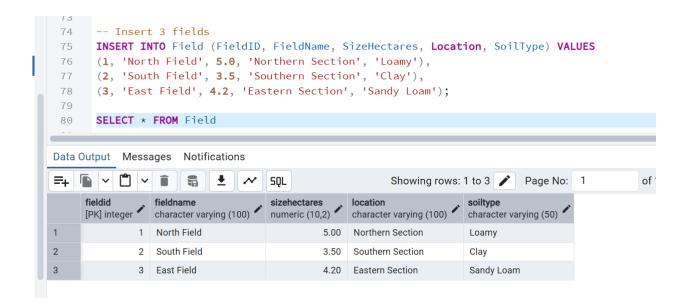
Explanation:

This populates the database with realistic test data:

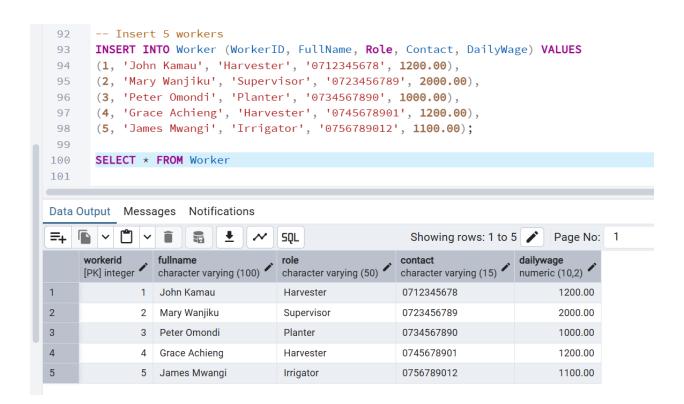
- 3 fields with different sizes and soil types
- 5 crops across different fields with various planting/harvest dates

- **5 workers** with different roles and wages
- Fertilizer applications showing input costs
- Harvest records tracking yield quantities and quality grades
- Sales transactions recording actual sales data

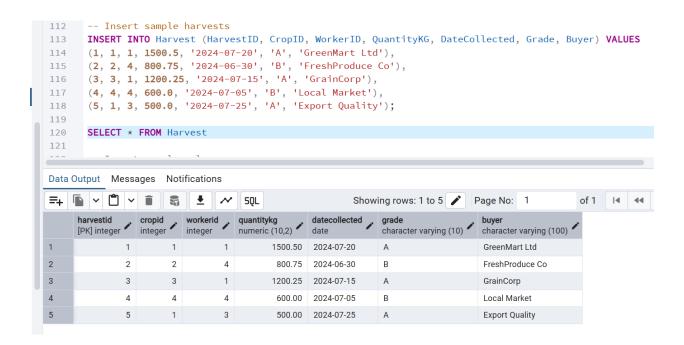
This data will allow us to test all subsequent queries effectively.

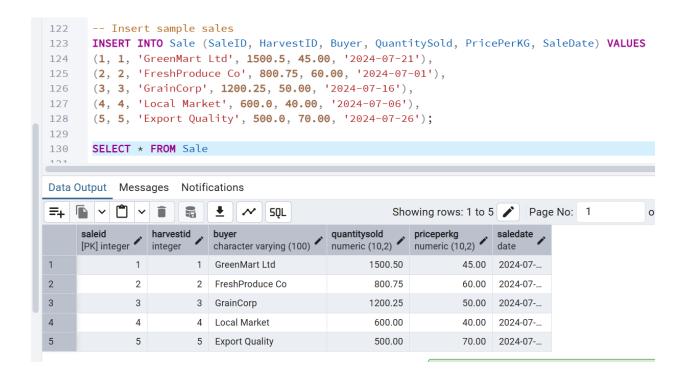


```
82 -- Insert 5 crops
 83
       INSERT INTO Crop (CropID, FieldID, CropName, PlantingDate, HarvestDate, Status) VALUES
       (1, 1, 'Maize', '2024-03-15', '2024-07-20', 'Harvested'),
 84
       (2, 1, 'Beans', '2024-04-01', '2024-06-30', 'Sold'),
 85
       (3, 2, 'Wheat', '2024-03-20', '2024-07-15', 'Harvested'),
 86
       (4, 2, 'Tomatoes', '2024-04-10', '2024-07-05', 'Ready'),
 87
       (5, 3, 'Potatoes', '2024-03-25', '2024-07-25', 'Growing');
 88
 89
 90
       SELECT * FROM Crop
 Ω1
Data Output Messages Notifications
=+ □ ∨ □ ∨
                             <u>*</u>
                                       SQL
                                                            Showing rows: 1 to 5
                                                                                    Page No: 1
                                                                                                         of
                                                plantingdate
                                                             harvestdate
                  fieldid
                           cropname
     [PK] integer
                 integer 🖍
                           character varying (100)
                                                date
                                                             date
                                                                         character varying (20)
1
                           Maize
                                                2024-03-15
                                                             2024-07-20
                                                                          Harvested
2
                           Beans
                                                2024-04-01
                                                             2024-06-30
                                                                          Sold
3
                           Wheat
                                                             2024-07-15
                                                                          Harvested
                                                2024-03-20
4
                        2 Tomatoes
                                                2024-04-10
                                                             2024-07-05
                                                                          Ready
               4
5
                           Potatoes
                                                2024-03-25
                                                             2024-07-25
                                                                          Growing
```



```
102 -- Insert sample fertilizers
         INSERT INTO Fertilizer (FertilizerID, CropID, Name, QuantityUsed, Cost, DateApplied) VALUES
  103
         (1, 1, 'NPK 17:17:17', 50.0, 2500.00, '2024-04-01'),
  104
         (2, 1, 'Urea', 25.0, 1500.00, '2024-05-15'),
  105
         (3, 2, 'DAP', 30.0, 1800.00, '2024-04-10'),
  106
         (4, 3, 'NPK 20:20:20', 40.0, 2200.00, '2024-04-05'),
  107
         (5, 4, 'Organic Compost', 60.0, 3000.00, '2024-04-20');
  108
  109
  110
         SELECT * FROM Fertilizer
  111
 Data Output Messages Notifications
 =+ □ ∨ □ ∨ ■
                              <u>*</u>
                                       SQL
                                                             Showing rows: 1 to 5 Page No: 1
                                                                                                         of 1
                         5
                                                                             dateapplied
                                                 quantityused
                                                               cost
       fertilizerid
                   cropid
                            name
                   integer
       [PK] integer
                            character varying (100) numeric (10,2)
                                                               numeric (10,2)
                                                                             date
 1
                          1 NPK 17:17:17
                                                         50.00
                                                                     2500.00
                                                                             2024-04-01
 2
                 2
                          1
                            Urea
                                                         25.00
                                                                      1500.00
                                                                             2024-05-15
  3
                 3
                             DAP
                                                         30.00
                                                                      1800.00
                                                                             2024-04-10
  4
                          3 NPK 20:20:20
                                                                      2200.00
                                                                             2024-04-05
                 4
                                                         40.00
 5
                          4 Organic Compost
                                                         60.00
                                                                      3000.00
                                                                             2024-04-20
```



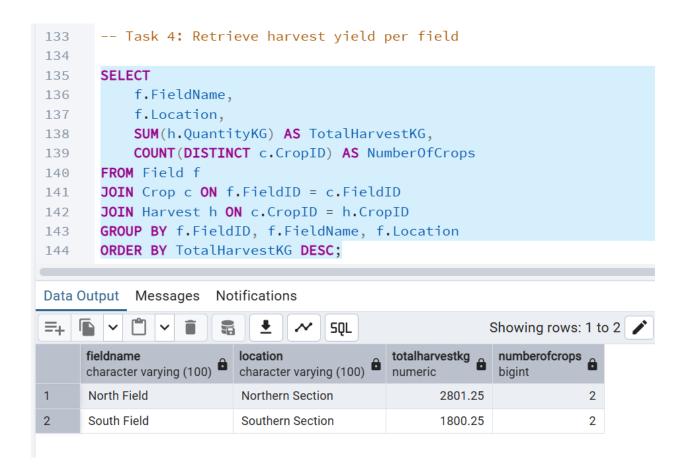


Task 4: Retrieve harvest yield per field

Explanation:

This query analyzes production efficiency by:

- SUM(h.QuantityKG) calculates total harvest weight per field
- COUNT(DISTINCT c.CropID) shows crop diversity per field
- **JOIN** operations connect Field \rightarrow Crop \rightarrow Harvest tables
- **GROUP BY** aggregates data at field level
- Results help identify most productive fields and optimize resource allocation

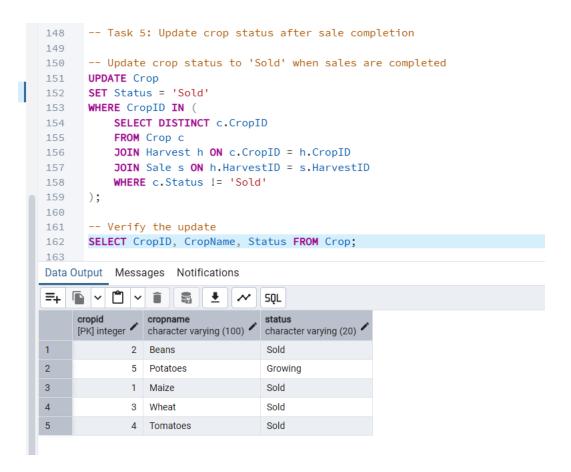


Task 5: Update crop status after sale completion

Explanation:

This automates business workflow by:

- Identifying crops that have been sold (have sales records)
- Updating their status from 'Harvested' or 'Ready' to 'Sold'
- Using a **subquery** to find crops with sales transactions
- Maintaining accurate inventory and production status
- Can be scheduled to run automatically after sales are recorded

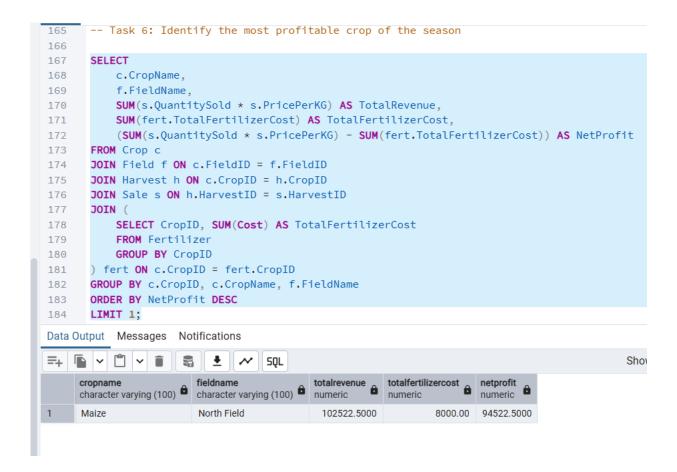


Task 6: Identify the most profitable crop of the season

Explanation:

This performs cost-benefit analysis by:

- Calculating **Total Revenue** (Quantity Sold × Price per KG)
- Calculating **Total Fertilizer Cost** from fertilizer applications
- Computing **Net Profit** (Revenue Fertilizer Cost)
- Using **subquery** to aggregate fertilizer costs per crop
- **ORDER BY NetProfit DESC** ranks crops by profitability
- **LIMIT 1** returns only the most profitable crop

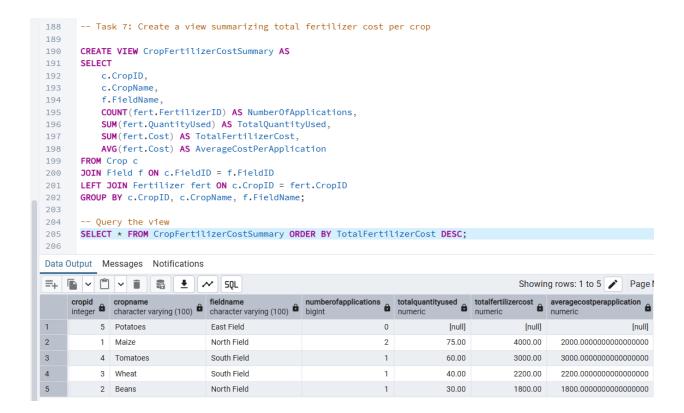


Task 7: Create a view summarizing total fertilizer cost per crop

Explanation:

This creates a reusable summary view that:

- **Aggregates fertilizer data** per crop (total cost, quantity, applications)
- Joins with field information for context
- Provides **pre-calculated metrics** like average cost per application
- Can be used in multiple reports without rewriting complex queries
- Updates automatically when underlying data changes



Task 8: Implement a trigger preventing fertilizer application before planting date

Explanation:

This ensures data quality by:

- BEFORE INSERT trigger validates data before saving
- Checks if fertilizer application date is before planting date
- Raises an error if validation fails using SIGNAL SQLSTATE
- Prevents illogical data (can't fertilize before planting)
- Maintains temporal consistency in farm operations

```
208 -- Task 8: Implement a trigger preventing fertilizer application before planting date
209
210
    CREATE OR REPLACE FUNCTION check_fertilizer_date()
211 RETURNS TRIGGER AS $$
212
    DECLARE
213
         plant_date DATE;
214 BEGIN
215
        -- Get the planting date for the crop
216
         SELECT PlantingDate INTO plant_date
217
         FROM Crop
218
         WHERE CropID = NEW.CropID;
219
220
         -- Check if fertilizer application date is before planting date
221 🗸
         IF NEW.DateApplied < plant_date THEN</pre>
222
             RAISE EXCEPTION 'Fertilizer cannot be applied before planting date';
223
         END IF;
224
         RETURN NEW;
225
226
     END:
$$ LANGUAGE plpgsql;
228
229
      -- Create the trigger
     CREATE TRIGGER check_fertilizer_date_trigger
230
      BEFORE INSERT ON Fertilizer
231
232 FOR EACH ROW
234
235
      -- Test the trigger (this should fail)
236 INSERT INTO Fertilizer (FertilizerID, CropID, Name, QuantityUsed, Cost, DateApplied)
237 VALUES (6, 1, 'Test Fertilizer', 10.0, 500.00, '2024-03-01');
Data Output Messages Notifications
```

ERROR: Fertilizer cannot be applied before planting date

CONTEXT: PL/pgSQL function check_fertilizer_date() line 12 at RAISE