Data Engineer Assessment - Warehouse Stock Management

Overview

Deadline: 1 hari dari waktu penerimaan
Expected Time: 3-4 jam
Submission: GitHub repository (private repo, invite reviewer)

Background

Anda diminta untuk membangun sistem data warehouse untuk mengelola stok gudang sebuah perusahaan e-commerce. Sistem ini harus dapat melacak pergerakan barang, mengelola multiple gudang, dan menyediakan analytics untuk decision making.

Task 1: Database Design (ERD)

Requirements

Buatlah ERD untuk sistem pengelolaan stok gudang dengan ketentuan:

Entities yang diperlukan:

- **warehouses** (gudang) - Multiple lokasi gudang
- **products** (produk) - Master data produk
- **categories** (kategori) - Kategori produk
- **suppliers** (supplier) - Data supplier
- **stock** (stok) - Current stock per gudang
- **stock_movements** (pergerakan stok) - History in/out
- **purchase_orders** (PO) - Order ke supplier
- **purchase_order_details** (Detail PO)
- **sales_orders** (SO) - Order dari customer
- **sales_order_details** (Detail SO)

Business Rules:

1. Satu produk bisa ada di multiple warehouses
2. Setiap movement harus tercatat (in/out/transfer/adjustment)
3. Track stock level, reorder point, dan safety stock
4. Support transfer antar gudang
5. Audit trail untuk semua perubahan

Deliverables

1. ERD diagram (PNG/PDF format) - bisa pakai draw.io, dbdiagram.io, atau tools lain
2. SQL DDL script untuk create semua tables dengan:
    - Primary keys dan foreign keys
    - Indexes yang diperlukan
    - Constraints (CHECK, UNIQUE, etc)
    - Comments untuk dokumentasi

File struktur:

/database/
- erd_diagram.png
- schema.sql
- README.md (penjelasan design choices)

Task 2: Generate Fake Data (1 Juta Data)

Requirements

Buatlah script Python untuk generate fake data dengan distribusi:

Data Volume:

- 10 warehouses
- 5,000 unique products
- 50 categories
- 200 suppliers
- 100,000 stock records (current stock)
- 500,000 stock movements (last 2 years)
- 100,000 purchase orders with ~300,000 detail lines
- 200,000 sales orders with ~600,000 detail lines

Data Characteristics:

- Realistic distribution (80/20 rule - 20% products generate 80% movements)
- Seasonal patterns untuk certain categories
- Random but logical data (e.g., stock movements affect current stock)
- Various movement types: 'IN', 'OUT', 'TRANSFER', 'ADJUSTMENT', 'RETURN'
- Some products should hit reorder point
- Include some data quality issues (5% records) untuk testing

Deliverables

Python script yang:

1. Generate semua fake data
2. Export ke CSV files atau direct insert ke database
3. Configurable parameters (dates, volumes, etc.)
4. Progress bar untuk monitoring
5. Data validation summary

File struktur:

```
/data_generator/
  - generate_data.py
  - requirements.txt
  - config.yaml
  - /output/ (CSV files atau SQL inserts)
```

Task 3: Database Functions & Stored Procedures

Requirements

Buatlah database functions/stored procedures untuk handle:

3.1 Stock Movement Function

```
-- Function untuk record stock movement dan update current stock
CREATE OR REPLACE FUNCTION record_stock_movement(
    p_product_id INT,
    p_warehouse_id INT,
    p_movement_type VARCHAR(20),
    p_quantity INT,
    p_reference_type VARCHAR(20),
    p_reference_id INT,
    p_notes TEXT
) RETURNS JSON
```

3.2 Stock Transfer Function

```
-- Function untuk transfer stock antar warehouse
CREATE OR REPLACE FUNCTION transfer_stock(
    p_product_id INT,
    p_from_warehouse_id INT,
    p_to_warehouse_id INT,
    p_quantity INT,
    p_notes TEXT
) RETURNS JSON
```

3.3 Reorder Alert Function

```
-- Function untuk check products yang perlu reorder
CREATE OR REPLACE FUNCTION check_reorder_points(
    p_warehouse_id INT DEFAULT NULL
) RETURNS TABLE(...)
```

3.4 Stock Valuation Function

```
-- Function untuk calculate stock value (FIFO/LIFO/Average)
CREATE OR REPLACE FUNCTION calculate_stock_value(
    p_method VARCHAR(10) -- 'FIFO', 'LIFO', 'AVG'
) RETURNS TABLE(...)
```

3.5 Audit Trigger

```
-- Trigger untuk audit trail semua changes
CREATE OR REPLACE FUNCTION audit_stock_changes()
RETURNS TRIGGER
```

Deliverables

1. SQL file dengan semua functions dan triggers
2. Test cases untuk setiap function
3. Documentation untuk penggunaan

File struktur:

```
/database_functions/
  - functions.sql
  - triggers.sql
  - test_cases.sql
  - README.md
```

Task 4: ETL Pipeline untuk Analytics

Requirements

Buatlah ETL pipeline dalam Python untuk generate analytics dan statistics:

4.1 Extract

- Connect ke database atau read dari CSV
- Implement incremental load (based on last update timestamp)
- Handle data quality issues

4.2 Transform - Calculate Statistics:

- **Inventory Metrics:**
  - Stock turnover ratio per product/category
  - Days of inventory on hand
  - Stock accuracy (physical vs system)
  - Dead stock identification (no movement >180 days)
- **Movement Analytics:**
  - Average daily movement per product
  - Peak periods identification
  - Movement trends (daily, weekly, monthly)
  - Seasonal patterns detection
- **Warehouse Performance:**
  - Utilization rate per warehouse
  - In/Out efficiency
  - Transfer patterns between warehouses
  - Geographic distribution optimization
- **Financial Metrics:**
  - Inventory value over time
  - Holding cost calculation
  - Stock-out cost estimation
  - ABC analysis (Pareto classification)

4.3 Load

- Create summary tables/materialized views
- Export to analytics-ready format (Parquet/CSV)
- Generate automated reports (PDF/HTML)

Deliverables

Python ETL pipeline dengan:

1. Modular design (separate E, T, L)
2. Configuration file untuk parameters
3. Logging dan error handling

4. Performance optimization untuk large datasets
5. Unit tests untuk transformation logic
6. Scheduling ready (dapat di-run via cron/airflow)

File struktur:

```
/etl_pipeline/
  - main.py
  - /extract/
    - data_extractor.py
  - /transform/
    - inventory_metrics.py
    - movement_analytics.py
    - warehouse_performance.py
    - financial_metrics.py
  - /load/
    - data_loader.py
    - report_generator.py
  - /tests/
    - test_transformations.py
  - /config/
    - config.yaml
  - requirements.txt
  - README.md
```

Task 5: Version Control & Documentation

Requirements

5.1 GitHub Repository Structure:

```
warehouse-stock-management/
├── README.md (main documentation)
├── .gitignore
├── requirements.txt (global)
├── /database/
│   ├── erd_diagram.png
│   ├── schema.sql
│   └── README.md
├── /database_functions/
│   ├── functions.sql
│   ├── triggers.sql
│   ├── test_cases.sql
│   └── README.md
├── /data_generator/
│   ├── generate_data.py
│   ├── requirements.txt
│   ├── config.yaml
│   └── /output/
├── /etl_pipeline/
│   ├── main.py
│   ├── /extract/
│   ├── /transform/
│   ├── /load/
│   ├── /tests/
│   ├── /config/
│   └── README.md
├── /analytics/
│   └── sample_reports/
└── /docs/
    ├── setup_guide.md
    ├── api_documentation.md
    └── performance_analysis.md
```

5.2 Main README.md harus include:

- Project overview
- Tech stack yang digunakan
- Setup instructions (step by step)
- How to run each component
- Sample outputs/results
- Performance considerations
- Future improvements
- Challenges faced dan solutions

5.3 Git Commits:

- Meaningful commit messages
- Logical commit progression
- Branch strategy (if any)