EXERCISE 5 – Advanced Data Transformation Using Pig

-- Load the product data

product\_data = LOAD 'product\_data.csv' USING PigStorage(',')

AS (product\_id:chararray, product\_name:chararray, price:float, category:chararray);

-- Load the sales data

sales\_data = LOAD 'sales\_data.csv' USING PigStorage(',')

AS (order\_id:int, product\_id:chararray, quantity:int, order\_date:chararray);

-- Filter products in the 'Electronics' category

filtered\_products = FILTER product\_data BY category == 'Electronics';

-- Group sales data by product\_id

grouped\_sales = GROUP sales\_data BY product\_id;

-- Join filtered products with grouped sales

joined\_data = JOIN filtered\_products BY product\_id, grouped\_sales BY product\_id;

-- Calculate total sales per product

total\_sales = FOREACH joined\_data GENERATE

filtered\_products::product\_name,

SUM(sales\_data.quantity \* filtered\_products.price) AS total\_sales;

-- Store the result in a new dataset

STORE total\_sales INTO 'total\_sales\_data' USING PigStorage(',');

5.1 --- -----------------------------------------

-- Load the datasets

products = LOAD 'products.csv' USING PigStorage(',')

AS (product\_id:int, product\_name:chararray, category:chararray, price:float);

sales = LOAD 'sales.csv' USING PigStorage(',')

AS (sale\_id:int, product\_id:int, quantity:int);

-- Calculate total revenue per sale (quantity \* price)

sales\_revenue = JOIN sales BY product\_id, products BY product\_id;

sales\_revenue = FOREACH sales\_revenue GENERATE

sales::sale\_id AS sale\_id,

products::product\_id AS product\_id,

products::category AS category,

products::price AS price,

sales::quantity AS quantity,

(products::price \* sales::quantity) AS total\_revenue;

-- Filter sales for high-value products (price >= 500)

high\_value\_sales = FILTER sales\_revenue BY price >= 500;

-- Group sales by category and calculate total quantity and revenue

category\_sales = GROUP sales\_revenue BY category;

category\_summary = FOREACH category\_sales GENERATE

group AS category,

SUM(sales\_revenue.quantity) AS total\_quantity,

SUM(sales\_revenue.total\_revenue) AS total\_revenue;

-- Join products and sales data based on product\_id

enriched\_sales = JOIN sales BY product\_id, products BY product\_id;

-- Calculate total revenue per product and sort by revenue

product\_revenue = FOREACH (GROUP sales\_revenue BY product\_id) GENERATE

group AS product\_id,

SUM(sales\_revenue.total\_revenue) AS total\_revenue;

sorted\_products = ORDER product\_revenue BY total\_revenue DESC;

-- Extract the top 3 products by revenue

top\_3\_products = LIMIT sorted\_products 3;

-- Filter product categories with total revenue > 1000

high\_revenue\_categories = FILTER category\_summary BY total\_revenue > 1000;

-- Display results

DUMP top\_3\_products;

DUMP high\_revenue\_categories;