ASSIGNMENT 3

1 Completed Sales Orders (Physical Items)

Business Problem:
Merchants need to track only physical items (requiring shipping and fulfillment) for logistics and shipping-cost analysis.

Fields to Retrieve:

- `ORDER_ID`
- `ORDER_ITEM_SEQ_ID`
- `PRODUCT_ID`
- `PRODUCT_TYPE_ID`
- `SALES_CHANNEL_ENUM_ID`
- `ORDER_DATE`
- `ENTRY_DATE`
- `STATUS_ID`
- `STATUS_DATETIME`
- `ORDER_TYPE_ID`
- `PRODUCT_STORE_ID`

`Solution:`

```
SELECT
        oi.ORDER_ID,
        oi.ORDER_ITEM_SEQ_ID,
        oi.PRODUCT_ID,
        pt.PRODUCT_TYPE_ID,
        oh.SALES_CHANNEL_ENUM_ID,
        oh.ORDER_DATE,
        oh.ENTRY_DATE,
        oi.STATUS_ID,
        os.STATUS_DATETIME,
        oh.ORDER_TYPE_ID,
        oh.PRODUCT_STORE_ID
FROM
        ORDER_ITEM oi
JOIN
        ORDER_HEADER oh ON oi.ORDER_ID = oh.ORDER_ID
JOIN
ORDER_STATUS os ON oh.status_id= os.status_id
```

```
JOIN
        PRODUCT p ON p.PRODUCT_ID = oi.PRODUCT_ID
JOIN
        Product_TYPE  pt On pt.product_type_id =p.product_type_id
WHERE
        oh.STATUS_ID = 'ORDER_COMPLETED'
        AND pt.is_Physical = 'Y' ;
```

2 Completed Return Items

Business Problem:
Customer service and finance often need insights into **returned items** to manage refunds, replacements, and inventory restocking.

Fields to Retrieve:

- `RETURN_ID`
- `ORDER_ID`
- `PRODUCT_STORE_ID`
- `STATUS_DATETIME`
- `ORDER_NAME`
- `FROM_PARTY_ID`
- `RETURN_DATE`
- `ENTRY_DATE`
- `RETURN_CHANNEL_ENUM_ID`

`Solution:`

```
SELECT
        rh.return_id,
        oh.order_ID,
        oh.product_store_id,
        rs.status_datetime,
        oh.order_name,
        rh.from_party_id,
        rh.entry_date,
        rh.return_date,
        rh.return_Channel_Enum_id
        FROM
        RETURN_HEADER rh  JOIN RETURN_ITEM ri
        ON rh.return_id= ri.return_id
        JOIN Order_Header oh  ON  oh.order_id=ri.order_id
        JOIN  Return_Status rs On rs.return_id=rh.return_id
        WHERE rh.status_id="RETURN_COMPLETED";
```

3 Single-Return Orders (Last Month)

Business Problem:
The mechandising team needs a list of orders that only have one return.

Fields to Retrieve:

- `PARTY_ID`
- `FIRST_NAME`

`Solution:`

```
SELECT
        rh.from_party_id As Party_id,
        p.first_Name
        From
        return_item ri JOIN return_header rh
        ON ri.return_id = rh.return_id
        JOIN  ORDER_HEADER oh ON
        oh.order_id=ri.order_id
        JOIN  Person p ON
        rh.from_party_id=p.party_id
        GROUP BY
        rh.from_party_id, p.FIRST_NAME
        HAVING
        COUNT(rh.RETURN_ID) = 1;
```

4 Returns and Appeasements

Business Problem:
The retailer needs the total amount of items, were returned as well as how many appeasements were issued.

Fields to Retrieve:

- `TOTAL RETURNS`
- `RETURN $ TOTAL`
- `TOTAL APPEASEMENTS`
- `APPEASEMENTS $ TOTAL`

`Solution:`

```
SELECT
        COUNT(ri.return_id) As TOTAL_RETURNS,
        SUM(ri.return_price*ri.return_quantity) As RETURN_TOTAL,
        COUNT(ra.return_Adjustment_type_id) AS TOTAL_APPEASEMENTS,
        SUM(ra.amount) AS APPEASEMENT_TOTAL
        FROM RETURN_ADJUSTMENT ra JOIN
        RETURN_ITEM  ri ON ra.return_id=ri.return_id
        WHERE ra.return_adjustment_type_id='APPEASEMENT';
```

5 Detailed Return Information

Business Problem:
Certain teams need granular return data (reason, date, refund amount) for analyzing return rates, identifying recurring issues, or updating policies.

Fields to Retrieve:

- `RETURN_ID`
- `ENTRY_DATE`
- `RETURN_ADJUSTMENT_TYPE_ID` (refund type, store credit, etc.)
- `AMOUNT`
- `COMMENTS`
- `ORDER_ID`
- `ORDER_DATE`
- `RETURN_DATE`
- `PRODUCT_STORE_ID`

`Solution:`

```
    SELECT
        rh.return_id,
        rh.entry_date,
        ra.comments,
        ri.order_id,
        ra.amount,
        oh.product_store_id,
        oh.order_date,
        rh.return_date,
        ra.return_adjustment_type_id
        FROM
        RETURN_HEADER rh JOIN
        RETURN_ITEM ri ON rh.return_id=ri.return_id
        JOIN order_header oh ON oh.order_id= ri.order_id
        JOIN RETURN_ADJUSTMENT ra ON ra.return_id=ri.return_id;
```

6 Orders with Multiple Returns

Business Problem:
Analyzing orders with multiple returns can identify potential fraud, chronic issues with certain items, or inconsistent shipping processes.

Fields to Retrieve:

- `ORDER_ID`
- `RETURN_ID`
- `RETURN_DATE`
- `RETURN_REASON`
- `RETURN_QUANTITY`

`Solution:`

```
SELECT
        ri.order_id,
        rh.return_id,
        rh.return_date,
        ri.return_quantity,
        ri.reason As RETURN_REASON
        From
        return_item ri JOIN return_header rh
        ON ri.return_id = rh.return_id
        GROUP BY rh.return_id,ri.order_id,rh.return_date,ri.reason,ri.return_quantity
        HAVING COUNT(rh.return_id)!= 1;
```

7 Store with Most One-Day Shipped Orders (Last Month)

Business Problem:
Identify which facility (store) handled the highest volume of “one-day shipping” orders in the previous month, useful for operational benchmarking.

Fields to Retrieve:

- `FACILITY_ID`
- `FACILITY_NAME`
- `TOTAL_ONE_DAY_SHIP_ORDERS`
- `REPORTING_PERIOD`

`Solution:`

```
SELECT
        oisg.facility_id,
         f.facility_name,
        count(oisg.order_id ) as TotalOneDayShippingOrder
from Order_Item_Ship_Group oisg
JOIN Order_shipment os on os.order_id = oisg.order_id
JOIN Shipment s on s.shipment_id = os.shipment_id
JOIN Facility f on f.facility_id = oisg.facility_id
where oisg.shipment_method_type_id = 'NEXT_DAY' and s.status_id = 'SHIPMENT_SHIPPED'
GROUP BY oisg.facility_id;
```

8 List of Warehouse Pickers

Business Problem:
Warehouse managers need a list of employees responsible for picking and packing orders to manage shifts, productivity, and training needs.

Fields to Retrieve:

- `PARTY_ID` (or Employee ID)
- `NAME` (First/Last)
- `ROLE_TYPE_ID` (e.g., “WAREHOUSE_PICKER”)
- `FACILITY_ID` (assigned warehouse)
- `STATUS` (active or inactive employee)

`Solution:`

```
SELECT
      pl.facility_id,
       pr.party_id,
       pr.role_type_id,
       pe.first_name || ' '  || pe.last_name as Full_Name,
       pty.status_id
from picklist pl
JOIN picklist_role pr on pr.picklist_id = pl.picklist_id
JOIN person pe on pe.party_id = pr.party_id
JOIN party pty on pty.party_id = pr.party_id;
```

---

9 Total Facilities That Sell the Product

Business Problem:
Retailers want to see how many (and which) facilities (stores, warehouses, virtual sites) currently offer a product for sale.

Fields to Retrieve:

- `PRODUCT_ID`
- `PRODUCT_NAME` (or `INTERNAL_NAME`)
- `FACILITY_COUNT` (number of facilities selling the product)
- (Optionally) a **list of FACILITY_IDs** if more detail is needed

`Solution:`

```
SELECT
  pf.product_id,
  p.INTERNAL_NAME AS PRODUCT_NAME,
  count(p.facility_id) AS FACILITY_COUNT
  FROM PRODUCT P JOIN PRODUCT_FACILITY pf
  ON pf.product_id=p.product_id
  GROUP BY pf.product_id,pf.facility_id;
```

---

10 Total Items in Various Virtual Facilities

Business Problem:
Virtual facilities (such as online-only fulfillment centers) handle a different inventory process. The company wants a snapshot of total stock across these virtual locations.

Fields to Retrieve:

- `PRODUCT_ID`
- `FACILITY_ID`
- `FACILITY_TYPE_ID`
- `QOH` (Quantity on Hand)
- `ATP` (Available to Promise)

`Solution:`

```
SELECT
        ii.product_id,
        ii.facility_id,
        f.facility_type_id,
        ii.quantity_on_hand_total AS QOH,
        ii.available_to_promise_total AS ATP
from Inventory_Item ii
JOIN facility f on f.facility_id = ii.facility_id
where f.facility_type_id = 'VIRTUAL_FACILITY' or f.facility_type_id = 'CONFIGURATION';
```

11 Transfer Orders Without Inventory Reservation

Business Problem:
When transferring stock between facilities, the system should reserve inventory. If it isn't reserved, the transfer may fail or oversell.

Fields to Retrieve:

- `TRANSFER_ORDER_ID`
- `FROM_FACILITY_ID`
- `TO_FACILITY_ID`
- `PRODUCT_ID`
- `REQUESTED_QUANTITY`
- `RESERVED_QUANTITY`
- `TRANSFER_DATE`

- STATUS

Solution:

```
SELECT

  it.inventory_transfer_id,

  it.facility_id,

  it.facility_id_to,

  ii.product_id,

  l.quantity,

  ii.quantity_On_Hand_Total,

  it.send_date,

  it.status_id

FROM Inventory_Transfer it JOIN inventory_item ii ON
ii.inventory_item_id=it.inventory_item_id

JOIN lot l ON l.inventory_item_id=it,inventory_item_id
```

12 Orders Without Picklist

Business Problem:
A picklist is necessary for warehouse staff to gather items. Orders missing a picklist might be delayed and need attention.

Fields to Retrieve:

- ORDER_ID
- ORDER_DATE
- ORDER_STATUS
- FACILITY_ID
- DURATION (How long has the order been assigned at the facility)

Solution: