**Reports Done at Behtar(LMIH Technologies)**

Reporting Manager:

Ashish Bajpai.

Done by:

Bona Swathi

Business Analyst Intern

[swathibona@gmail.com](mailto:swathibona@gmail.com)

8762883698

**Contents:**

1. Ageing Report - warehouse & Store wise
2. Stock transfer from warehouse to store
3. Sales order returns
4. Fraud detection in offline sale
5. Duplicate additions and deductions of inventory
6. Count of multiple coupons
7. Returned items of multiple coupons applied
8. Order details of the items
9. Sale return with High price than purchase price

10 Inward & Outward

11 Inventory Gap- Op/stock+purchases-sales=derived c/stock -c/stock= var

12 Clear carts

13. Redash CleanUp

14. Category\_wise share in overall sales with ABV, Bill\_cuts

15. Current selling price details(after discounts)

16. Reports related to multiple Grn issue

1. Multiple Grn- Warehouse\_wise
2. Multiple store\_inventory changes
3. Multiple store\_inventory changes ---- only for stock transfermode
4. Multiple store\_inventory changes -----for GRN

17.Margin Store\_wise(Summary)

18. SplitBill Testing Query

19. SplitBill actual and total value query

20. Store\_wise contribution in sales, margin, bill\_cuts

21. Tested the Build\_1\_staging.apk and Build\_2\_staging.apk

22.splitbill actual and total value query

23.Duplicate Generated Bills

24.a) Warehouse\_inventory Audit Report

b) Store\_inventory Audit report

25) Stock Transfer Return

a)filterby Warehouse

b)filterby Invoice\_number

26)Fraud detection Reports

1. Multiple Times of Returning
2. Returning item which are not purchased
3. Returned\_value > ordered\_value

27) Warehouse\_inventory\_opening& closing

28) Behtar Inventory\_amount report

1. Warehouse\_wise
2. Store\_wise

29) Landing\_Price& Tax

30) Shrinkage Report

a)Store\_wise

b)Warehouse\_wise

31)Warehouseinventoryamount\_quater\_wise

32)Stock transfer with seller\_id and Supplier\_id

33) Sku\_correction

1. Sku\_correction with barcode, store\_id, cashier\_login
2. Sku\_correction with sp, mrp, unit

34)Top 50 SP with barcodes

35) Sales demand prediction model using python.

36) Margin by article

37) Articles not being sold at the store

1. where not\_sold for != null
2. where not\_sold for = null
3. **Ageing Report:**  Ageing and not being sold items and made to tell the importance of expiry\_date data.

[Ageing Report (redash.io)](https://app.redash.io/behtar1/dashboard/ageing-report)

**a)Warehouse wise:** This report contains the data of the items which are lying in the warehouse by not being sent to the store

Objective: To dispatch the items which are coming into the warehouse based on inwarding date or batch wise

Approach: Taken the Warehouse\_warehouseinventory(WWI) table(where the warehouse inventory data exists) left joined with order\_storewarehouseorderitem(OSWI -where the stock transfer data from warehouse to stock exists) and given a condition where sent\_qty is NULL, means, if sent\_qty is null then the items are not sent to any store and they are lying in the warehouse from the past days.

Results:

* In this report, we can separately check the ageing items in each warehouse and in each warehouse\_id, with the help of filters.
* This report contains the item\_name,article\_name, warehouse\_name&id, unit, grammage, article\_code, Barcode, along with the nonmovingdays- which tells from how many days the item is not moving from the warehouse.
* This report is for bangalore city.

Benefits:

* This Kind of data is helpful in dispatching the items fastly, before expiry\_date from the warehouse to store.
* This data helps in avoiding the goods damage because of expiry
* This helps in reducing the loss by dispatching the items fast before it gets expire

**b)Store Wise:** This report contains the data of the items which are lying in the store by not being sold.

Objective: To sell the items which are coming into the store against the batch or which are inwarded first into the store.

Approach: Taken the store\_storeinventory table, with the help of last\_sold column found from how many days the items are not selling from the store.

Results:

* In this report, we can separately check the ageing items in each store and in each store\_id with the help of filters.
* This report contains the item\_name,article\_name, store\_name&id, unit, grammage, article\_code, Barcode, along with the notsolddays- which tells from how many days the item is not selling from the store
* This report is for bangalore city.

Benefits:

* This Kind of data is helpful in selling the items fastly, before expiry\_date from the store.
* This data helps in avoiding the goods damage because of expiry
* This helps in reducing the loss by selling the items first which are expiring soon

**2. Stock Transfer from warehouse to store:** This report contains the data of the stock which are transferred from warehouse to store.

[Stock transfer from warehouse to store (redash.io)](https://app.redash.io/behtar1/dashboard/stock-transfer-from-warehouse-to-store?p_daterange=2020-10-01--2020-10-31)

Objective: To find the items which are transferring from warehouse to store.

Approach: Taken the order\_storewarehouseorder and order\_storewarehouseorderitem table by giving the condition sent\_qty>0, so that only sent items from warehouse to store will record.

Results: In this report, we can see from when the stock transferred from warehouse and when the stock reached to store with the warehouse and stock information and how much quantity is sent from warehouse to store.

Benefits : With this information, we will know how much stock is transferring from warehouse to store monthly, based on that, we can make the decisions how much stock we should purchase for the next month.

**3. Sales Order Returns:** This report shows the returns from the sales

[Sales order returns for october (redash.io)](https://app.redash.io/behtar1/dashboard/sales-order-returns-for-october?p_daterange=2020-11-01--2020-11-17)

Objective: to find the items which are returning from the sales

Approach: Taken order\_storeorder and order\_orderreturn tables and join sku\_items\_sku with the barcode, so that only the returned items will record.

Results: This report contains the details of the items along with the bill\_number, store\_name, how many no. of items are returned, its value and the article\_code.

Benefits: With this report, we can get the information of how many items are returning and we can check why they are returning, so that we can try to minimize them for the next month.

**4. Fraud Detection in offline sale:** This report contains the details of the items of fraud where the items sold\_quantity is > 10 in offline.

[Fraud detection for offline sale (redash.io)](https://app.redash.io/behtar1/dashboard/fraud-detection-for-offline-sale?p_DateRange=2020-05-01--2020-11-09)

Objective: To find fraud items in offline sales.

Approach:

* To find the offline sale items, given a condition like where mobile number is NULL in analytics\_storeorderitems(sales)
* Fraud usually happens in bulk, so given the condition quantity >10.

Results: This reports contains the sales items information in offline sale where the quantity>10, along with the store name, article\_code and barcode

Benefits: With this report we can check where the frauds are happening in offline sales and we can take steps to minimise those offline frauds.

**5.Multiple Additions and Deductions of Inventory:** Report to track multiple inventory additions or deductions on a daily basis.

[Duplicate deductions and additions of inventory (redash.io)](https://app.redash.io/behtar1/queries/546593/source?p_daterange=2020-08-01--2020-11-26)

Approach: To Find the duplicate rows which are entered multiple times, snapshot\_storeinventoryarticlecodechanges table was taken and used the COUNT function to find the duplicate rows.

Results:

* This report contains the details of the items which are entered multiple times, along with the information of invoice, article\_code, mode, countof(how many no of duplicate rows).
* The invoices of storeorderdeletion and storeauditcleanup is null because in the data itself, they are null.

**6. Count of Multiple Coupons**: This report contains the details of the items where the coupons are applied multiple times with the help of coupon\_applied column and count function.

[count of multiple coupons (redash.io)](https://app.redash.io/behtar1/queries/520183)

**7. Returned items of multiple coupons applied:**

[multiple coupons returns (redash.io)](https://app.redash.io/behtar1/queries/521272/source)

Objective: This report is mainly done to find the items or people who have applied multiple coupons while purchasing and returning the items with the intention of crediting the extra amount into their accounts.

Approach: Taken the order\_storeorder and order\_orderreturn table and used the coupon\_applied column and count function to find the returned items where the coupons are applied multiple times.

Results:

* The report is showing the returned items with the count of coupons applied along with bill\_number.

Benefits : we can check how many coupons are applying and how much of money we are losing with the multiple coupons and we can take actions to minimize the loss.

**8. Order Details of the items:** This report contains the order details of the last one year

[order details with barcode from oct 2019 to oct 2020 (redash.io)](https://app.redash.io/behtar1/queries/506868/source)

**9. Sale Return with High Price than Purchase Price:** A daily report to identify the cases across all stores where an item was purchased at a price and was returned at a higher price(whether it might be bill buster, coupon applied, discount offered)

[sale return with high price (redash.io)](https://app.redash.io/behtar1/queries/563572/source?p_DateRange=2020-12-05--2020-12-05)

Objective: To find the items which are purchased at one price and returned at higher price. This report is mainly done to find where the fraud is happening like buying the items with discount, or coupon applied,or billbuster and returning the same items with the intention of crediting the extra amount into their accounts.

Approach: Taken the order\_returnorder, order\_storeorder and analyticsstoreorderitems tables by giving the filter of returnunitprice > orderunitprice.

Results: This report contains the details of the items where the returnunitprice>orderunitprice along with sold\_quantity, returnedquantity, sales\_sp, coupon applied or not.

Benefits: It helps in finding the frauds in returns

**10. Inward & Outward:** This report shows the inward & Outward inventory of the warehouse.

[Redash](https://app.redash.io/behtar1/dashboard/inward-outward-inventory-report)

1. **Inward:** This report contains the data of the stock which are coming into the warehouse daily

Approach: Used the grn\_goodsreceiptenote, grn\_grnitems, sku\_items\_sku tables to find the stock which is coming into the warehouse daily.

Results: Report contains the details of the grn data along with the filter for date and warehouse\_name.

Benefits: with this data, the operational and marketing team can make the comparison between how much the stock is coming into the warehouse and how much the stock is going out of the warehouse and they can take some decisions like how much stock needs to buy or sell for the next month.

1. **Outward:** This report contains the data of the stock which are going out of the warehouse daily.

Approach: Used the orde\_storewarehouseorder, order\_storewarehouseorderitem, sku\_items\_sku tables to find the stock which is going out of the warehouse daily.

Results: Report contains the details of the stock transfer data from warehouse to store along with the filter for date and warehouse\_name.

Benefits: with this data, the operational and marketing team can make the comparison between how much the stock is coming into the warehouse and how much the stock is going out of the warehouse and they can take some decisions like how much stock needs to buy or sell for the next month.

**11. Clear cart**

[clear cart (redash.io)](https://app.redash.io/behtar1/dashboard/clear-cart)

1. StoreID, Countof clearcarts with value zero, count of clearcarts with value > 0
2. Storeid, userlogin, count of clearcarts, in past 15 days
3. Storeid, cartvalue, clearcart, discount/coupon applied, in desc order for the last 24 hours.

**12. Inventory Gap: Op/stock inventory + purchase inventory-sales = derived c/stock -actual c/stock = var:** This report is mainly done to know the difference between actual closing stock and the derived closing stock, which helps in finding the variance in stock at the warehouse level and store level.

[Inventory Gap(Warehouse\_wise) (redash.io)](https://app.redash.io/behtar1/queries/572072/source?p_daterange=2020-11-01--2020-11-30)

****

1. **Warehouse\_wise:** This report contains the stock difference at warehouse level

Objective: To know the difference between derived closing inventory and closing inventory at warehouse level

Approach: Taken the opening and closing inventory from warehouse\_warehousesnapshotarticle article table, inward from purchase\_GRN table and Outward(outward sku movement which means the stock transferred from warehouse to store) from orderstorewarehouseorder table and applied a formula to find the stock difference.

(openingvalue+inward\_value-stocktransfer\_value)-warehouseactual\_closingvalue) as wareinventory\_difference.

Results:

* This report contains the openining\_stock, closing stock, inward, outward stock of the warehouse for every month or every day.

Benefits:

* With this data, we can see the difference between the actual and derived closing stock and can check where the difference is happening in the stock at warehouse level.

1. **Store\_wise :** This report contains the stock difference at Store level

Objective: To know the difference between derived closing inventory and closing inventory at storelevel.

Approach: Taken the opening and closing inventory from snapshot\_storeinventorysnapshotdaily, inward(stock transfer from warehouse to store) from order\_storewarehouseorder table and Outward(sales at store\_level) from analytics\_storeorderitemstable and applied a formula to find the stock difference.

(openingvalue+inward\_value-sales\_value)-storeactual\_closingvalue) as store\_difference.

Results:

* This report contains the openining\_stock, closing stock, inward, outward stock of the store for every month or every day.

Benefits:

* With this data, we can see the difference between the actual and derived closing stock and can check where the difference is happening in the stock at store level.

**13. Redash Clean Up:** Archived the queries which has the name of New Query, practice query and long queries and unpublished queries and queries which are not in use after confirming with the creator.

**14. Category\_wise share in Overall sales along with ABV and Bill\_Cuts:** This report contains the data of all stores each category share and category sales in overall sales along with AVB and Bill\_cuts.

[category share in overallsales with ABV, Bill\_cuts (redash.io)](https://app.redash.io/behtar1/queries/591390/source)

Approach: created subqueries for overall sales and sales by category\_wise and bill\_cuts, mainly using tables analytics\_storeorderitems, order\_storeorder and catlog\_toplevel tables.

Results : category\_wise sales, share % in overall sales in each store\_id along with ABV and Bill\_cuts.

Benefits : Through this, we can find which category is contributing more and which is less contributing to the sales and in which category ABV and Bill cuts are occurring more.

**15. Current Selling Price details(after discount):** This report made to know the current selling price, mrp and discount offers to the product.

[Current Selling Price details(after discounts) (redash.io)](https://app.redash.io/behtar1/queries/599657/source?p_daterange=2020-12-01--2020-12-31)

Approach: Used the tables sku\_items\_multiplemrpandsp and offers\_discountofferarticle to know the selling price and mrp for the products

* There are discount offers i.e. percentage and value, so used two different formulas for each to know the actual sold price after subtracting the discount from the selling price.
* discount\_type ='PERCENTAGE' THEN sp-(sp\*value/100)
* discount\_type = 'VALUE' THEN (sp-value) else 0

Results: The results were showing the city wise : article\_code and barcode wise selling price , mrp discount\_type and actualsellingprice(after removing discount value).

Benefits: The prices will change almost every day and this must be properly communicated to the teams and customers to ensure transparency in operations. This report will be helpful in these kinds of situations.

**16. Reports related to multiple Grn issue:**

1. Multiple Grn- Warehouse\_wise

<https://app.redash.io/behtar1/queries/602090>

1. Multiple store\_inventory changes

<https://app.redash.io/behtar1/queries/602834>

1. Multiple store\_inventory changes ---- only for stock transfermode

<https://app.redash.io/behtar1/queries/610062>

1. Multiple store\_inventory changes -----for GRN

<https://app.redash.io/behtar1/queries/611355>

**17. Margin Store\_wise(Summary):** This report is mainly done to transfer one of the report from tableau to redash:

[Margin-store\_wise(summary) (redash.io)](https://app.redash.io/behtar1/queries/616056/source?p_daterange=2021-01-01--2021-01-31)

Objective: To find the store\_wise Margin\_value, Margin Percent, sales and sales share of each store in overall sales(In the requirement format of tableau margin by store report)

Approach: Written the two sub-queries using the analytics\_storeorderitems and few other tables.

* 1st sub-query is for calculating the grandtotal of sales
* 2nd subquery is for calculating store\_wise sales, COGS, Margin and Margin percentage.
* Then written an outer query, by calculating the salesshare of each store in overall sales along with sales, margin and margin percentage of each store.

Results: Store\_wise Sales, Margin, Margin percentage and sales share of each store in overall sales (as per the tableau report )

**18. SplitBill actual Testing query:**

[split bill testing query (redash.io)](https://app.redash.io/behtar1/queries/637464/source)

Objective: This report is mainly done to test the split bill part in the Behtar App.

**19. SplitBill actual and total value query:**

[splitbill actual and total value query (redash.io)](https://app.redash.io/behtar1/queries/631644/source)

Objective**:** To find the products where the value of the product and the actual\_value(cash+upi+online) are not equal.

Approach**:** Extracted the order\_number, value, cash, upi, online, sum(cash+upi+online)

From orderstoreorder table and given the condition of Value!=Actualvalue

**20.Store\_wise contribution in sales, margin, bill\_cuts:**

[**Store\_wise contribution in sales, margin, bill cuts (redash.io)**](https://app.redash.io/behtar1/queries/621252/source?p_daterange=2021-01-01--2021-02-01)

Objective: To find the store\_wise sales, Margin, no.of bill cuts and each store contribution in sales and margin

Approach: Written the two sub-queries using the analytics\_storeorderitems and few other tables.

* 1st subquery is for calculating the grand total of sales and COGS
* 2nd subquery is for calculating store\_wise sales, COGS, Margin and Margin percentage.
* Then written an outer query, by calculating the sales share and margin of each store in overall sales and margin along with sales, Margin and No.of billcuts of each store.

Results: Store\_wise Sales, Margin, No of BillCuts, sales and Margin share of each store in overall sales.

**22.Splitbill actual and total value query:** This report was prepared to check for any differences between actual\_value and value\_cash+value\_online+value\_upi(where the payment happened through split bill) in the split Bills.

[splitbill actual and total value query (redash.io)](https://app.redash.io/behtar1/queries/631644/source)

**23.Duplicate Generated Bills:**

[Duplicate Generated Bills (redash.io)](https://app.redash.io/behtar1/queries/633713/source)

Objective: To find the duplicate generated bills having count>1

Approach: Used the order\_storeorder and analytics\_storeorderitems Table with is\_duplicate=’false’ condition

**24. Inventory Audit Report:**

1. **Warehouse\_Audit\_inventory Report:**

[Warehouse\_inventory Audit Report (redash.io)](https://app.redash.io/behtar1/queries/677023/source?p_daterange=2021-02-23--2021-02-23)

Used Tables: Snapshot\_warehouseinventorychanges

Columns: Mode, Date, Warehouse\_Id, Warehouse\_name, City, barcode, SKU\_Title, from Qty, To\_quantity, Invoice\_id, User\_id, user\_Name

Filters: City, Mode, User\_name, Warehouse\_name

1. **Store\_Audit\_Inventory Report:**

[**Store Inventory Audit Report (redash.io)**](https://app.redash.io/behtar1/queries/676980/source?p_daterange=2021-02-23--2021-02-23)

Used Tables: Snapshot\_warehouseinventorychanges

Columns: Mode, Date, Store\_Id, Store\_name, City, barcode, SKU\_Title, from Qty, To\_quantity, Invoice\_id, User\_id, user\_Name

Filters: City, Mode, User\_name, Store\_name

**25) Stock Transfer Return:** Returns of the stock transfer filtered by Warehouse\_nameand invoice\_number

Objective: to find the items which are returned from store to warehouse

Approach: used the storewarehousereverseorder and order\_storewarehouseorder tables to find the items which are returned from store with barcodes, invoice\_numbers and stock\_transfer\_ids

**a)Stock transfer\_return with warehouse\_name filter:** Applied Warehouse\_name filter

Columns: Date, Invoice\_number, Stock\_transfer\_id, total\_value, sku\_name, barcode, article\_code, warehouse\_name, warehouse\_id, store\_name, store\_id, city\_id

[Stock Transfer Return--filtered by Warehouse (redash.io)](https://app.redash.io/behtar1/queries/684934/source?p_daterange=2021-02-01--2021-02-28)

**b)Stock transfer\_return with invoice\_number filter:** Applied Invoice\_number filter

Columns: Date, Invoice\_number, total\_value, quantity, sku\_id, sku\_name, barcode, article\_code, city\_id

[**StockTransferReturn--filterby Invoice\_number (redash.io)**](https://app.redash.io/behtar1/queries/684874/source?p_daterange=2021-02-01--2021-02-28)

**26)Fraud detection Reports:** These are mainly done to find the frauds happening at the organization in different situations.

1. **Multiple Times of Returning:** Returning the same item Multiple times

[multiple times of returning (redash.io)](https://app.redash.io/behtar1/queries/685099/source?p_DateRange=2021-01-01--2021-01-31)

Approach: Using Order\_returnorder and order\_storeorder tables and applied the condition of count(ORDERBARCODE)>1 so that only multiple returned items can be extracted.

Columns: Returned\_on, order\_id, total\_returned\_items, order\_number, Order\_barcode, count(orderbarcode), returned\_barcode, returned\_value, ordered\_value, store\_name, cashiers(mobile\_numbers).

1. **Returning item which are not purchased:** Returning the items which are not purchased by the customer

[Returning item which are not purchased (redash.io)](https://app.redash.io/behtar1/queries/688636/source?p_DateRange=2021-01-01--2021-01-31)

Approach: Using Order\_returnorder and order\_storeorder tables and applied the condition of Returnedbarcode != ORDERBARCODE, so that only returned items which are not ordered or purchased

Columns: Returned\_on, order\_id, total\_returned\_items, order\_number, Order\_barcode, returned\_barcode, returned\_value, ordered\_value, store\_name, cashiers(mobile\_numbers).

1. **Returned\_value > ordered\_value:** Returned items where the returned\_value more than the ordered\_value

[Returned\_value> ordered\_value (redash.io)](https://app.redash.io/behtar1/queries/688619/source?p_DateRange=2021-01-01--2021-01-31)

Approach: Using Order\_returnorder and order\_storeorder tables

Columns: Returned\_on, order\_id, total\_returned\_items, Order\_barcode, returned\_value, ordered\_value, extrareturned\_amount, order\_number, store\_name, cashiers(mobile\_numbers).

**27) Warehouse\_inventory\_opening& closing:**

[Inventory-warehouse-opening&Closing (redash.io)](https://app.redash.io/behtar1/queries/709774/source)

Objective: To find the opening & closing Inventory value of each warehouse on each day.

Approach: Used the warehouse\_warehousesnapshotarticle for opening and closing quantity and sku for selling price(SP).

**28) Behtar Inventory\_amount report:** To find the overall Inventory value present at warehouse and store level of each article

1. Inventory\_amount\_Warehouse\_wise:

[Inventory\_amount- warehouse\_wise (redash.io)](https://app.redash.io/behtar1/queries/696115/source?p_DateRange=2021-03-05--2021-03-05)

Used\_tables: Warehouse\_warehouseinventory and sku by warehouse\_name filter

Columns: created\_on, city\_id, arcticle\_name, L1, L2, Warehouse\_id, Warehouse\_name,value, wh\_qty.

1. Inventory\_amount\_store\_wise:

[Inventory\_amount- Store\_wise (redash.io)](https://app.redash.io/behtar1/queries/705592/source?p_DateRange=2021-03-05--2021-03-05)

Used\_tables: store\_storeinventory and sku by store\_name filter

Columns: date, city\_id, arctile\_name, L1, L2, store\_id, store\_name,value, store\_qty.

**29)Landing\_Price& Tax:** To find the clear difference between basic\_price, tax\_value, price\_with\_gst.

[LP & Tax (redash.io)](https://app.redash.io/behtar1/queries/709082/source?p_DateRange=2021-03-08--2021-03-08)

Used\_tables: Warehouse\_warehouse, grn tables.

Columns: sku\_name, article\_name, lo\_top\_level, l1\_section, l2\_category, l3\_subcategory, mrp, sp, basic\_price, tax\_value, quantity, basic\_total, tax\_total, price\_with\_gst, avg\_landing\_price, brand, company\_name.

**30) Shrinkage Report:** This report is mainly done to find the shrinkage\_value (Like damage ) at store and warehouse\_level.

**a)Store\_wise:**To find the items which are shrinkage wasting and expiring at store\_level.

[Shrinkage Report-- Store\_wise (redash.io)](https://app.redash.io/behtar1/queries/713771/source?p_Date%20Range=2021-02-01--2021-02-28)

Approach: Using snapshot\_storeinventorychangesand sku\_items\_sku tables and applied the condition of mode ='Expiry and Damage’.

Columns: date, store\_id, store, shrinkage\_quantity, sp, total\_value, mode, section, category, sku\_name

**b)Warehouse\_wise:**

Objective: To find the items which are (shrinkage) wasting and expiring at store\_level.

[shrinkage-warehouse\_wise (redash.io)](https://app.redash.io/behtar1/queries/715415/source?p_Date%20Range=2021-02-01--2021-02-28)

Approach: Using warehouse\_warehouseskucorrectionitems and sku\_items\_sku tables and applied the condition of option ='Sinkage'

Columns: date, item\_total, quantity, option, section, category, sku\_name

**31)Warehouseinventoryamount\_quater\_wise:** To find the total\_value and total\_quantity of each warehouse for a quarter.

[warehouseinventoryamount\_quater\_wise (redash.io)](https://app.redash.io/behtar1/queries/715561/source?p_DateRange=2021-01-01--2021-03-16)

Approach: Using warehouse\_warehouseinventory and sku\_items\_sku tables

Columns: warehouse\_id, warehouse\_name, value, warehouse\_quantity of each warehouse for a quarter.

**32). StockTransfer with seller\_id and supplier\_id:** This is mainly prepared to know the stock transfer with supplier\_id

[Stock Transfer with seller\_id and supplier\_id (redash.io)](https://app.redash.io/behtar1/queries/724273/source?p_daterange=2021-03-23--2021-03-23)

Approach: From purchase\_orders\_purchaseorderitems table taken the supplier\_id column and left joined with stock transfer query.

Benefits: In this report we can see the stock transfer from warehouse to store along with the supplier\_id and seller\_id.

Columns: Invoice\_number, article\_code, seller\_id, supplier\_id, supplier\_name, sku\_id, sent\_qty, sent\_value, warehouse\_name, store\_name.

**33). Sku Correction:** This report is prepared to find the new skus which are added newly after march 31st 2021.

1. **Sku\_correction with barcode, store\_id, cashier\_login**

[sku\_correction with store\_id, barcode, cashierlogin (redash.io)](https://app.redash.io/behtar1/queries/729142/source)

Approach: Used analytics\_analyticsevents from productionsnapshot table, excluded the dummy store(store\_id =5) and considered the new skus only after march 31st 2021, by giving the condition of event\_name ='add\_sku'

Columns: created\_on, event\_name, store\_id, barcode, cashier\_login.

1. **Sku\_correction with sp, mrp , unit**

[sku\_correction with sp, mrp, unit (redash.io)](https://app.redash.io/behtar1/queries/732554/source#1225665)

Approach: Used analytics\_analyticsevents from productionsnapshot table, excluded the dummy store(store\_id =5) and considered the new skus only after march 31st 2021, by giving the condition of event\_name ='add\_sku'

Columns: created\_on, barcode, count(barcode), sp, mrp, unit, event\_name, store\_id, sku\_title, weightage, total\_payable\_value and cashier\_login.

**34). Top 50 SP with barcodes :** This report is prepared to find the top high SP with the barcodes.

[Top 50 SP with barcodes (redash.io)](https://app.redash.io/behtar1/queries/737313/source)

.

Approach: Used sku\_items\_sku table and extracted the values where sp is not null.

Columns: city\_id, sp, mrp, barcode.

**35) Sales demand prediction model using python:** Build this model, mainly to predict the next 3 month sales

[salesprediction - Colaboratory (google.com)](https://colab.research.google.com/drive/1JrNufNYIcRgF0q2GRwfqObCNEZmwniT1?usp=sharing#scrollTo=suitable-beads)

Approach:

a)Data: The data was taken from feb 1st to 19th april which contains (1695988 rows and 9 columns) by writing a query in redash.

[sales data for demand production model 2 (redash.io)](https://app.redash.io/behtar1/queries/748439/source?p_DateRange=2021-02-01--2021-04-19)

Columns: store\_id, store\_name, date, day of week, day\_name, barcode, quantity, sp

b)processing: some findings from the data

* Sales trend over days - on which day sales occur more ?
* Which store\_id has more sales ?
* Which barcode has more sales ?

c) Pre-Processing of the data: Removed the not required variables of store\_name, day\_name, date, day of week, sp

* To build the Multiple linear regression model, all the variables should be in float, so converted the barcode to float
* Removed the null variables
* After removing the null variables the data contains (1515457 rows, 4 columns)

d) correlation between dependent variable(sales) and independent variable

sales 1.000000

quantity 0.151901

barcode -0.005886

store\_id -0.007971

e) Model Building:

* Splitted the dataset in 80:20 ratio
* Build the multiple linear regression model

f) Results: R2 is 0.03829,

Model got very less R2 and model is not able to predict the matching or exact sales.

A low R-squared value indicates that our independent variable is not explaining much in the variation of our dependent.

**36) Margin by article:** This report is done to find the margin by article wise

[margin by article (redash.io)](https://app.redash.io/behtar1/queries/762504/source?p_daterange=2021-05-01--2021-05-02)

Columns: article\_code, article\_name, Brand, total\_quantity\_sold, weighted\_avg\_sp, total\_sales\_value, m\_avg\_lp, cogs, margin, md\_margin

**37) Articles not being sold at the store:** This report is prepared to find the articles which are not being sold from some days at the store

Columns: store\_name, store\_id, article\_code, barcode, quantity, sp, overvalue, article\_name, sku\_name, last\_sold, not\_sold\_for,

Approach: Taken the store\_storeinventory table, with the help of last\_sold column found from how many days the items are not selling from the store.

Not\_sold\_for has null values and not null values, so splitted this into two reports.

1. **where not\_sold for != null:** To find the articles where not \_sold\_for column is not null by giving not\_sold\_for != null condition

[articles not being sold at the store where not\_soldfor != null (redash.io)](https://app.redash.io/behtar1/queries/763471/source)

1. **where not\_sold for = null:** To find the articles where not \_sold\_for column is null by giving not\_sold\_for= null condition

[articles not being sold at the store -- where not\_soldfor = null (redash.io)](https://app.redash.io/behtar1/queries/764010/source)