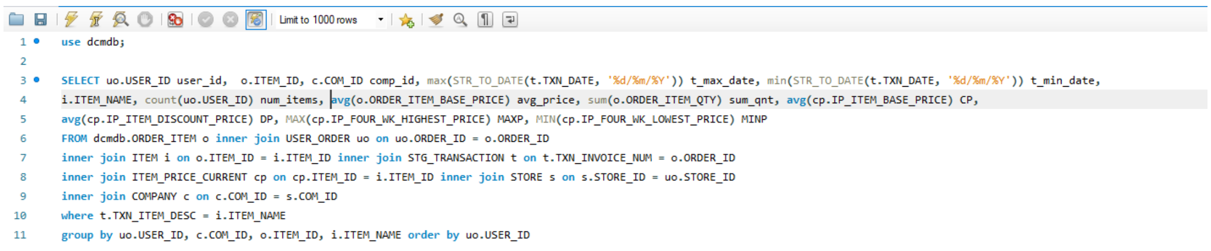
Developing a Content-based Filtering approach for item recommender

- **Advantages of Content-based Filtering approach**
  - Content-based filtering uses item features to recommend other items like what the user likes, based on their previous actions or explicit feedback.
  - Content-based filtering does not require other users' data during recommendations to one user.
  - The model can capture the specific interests of a user, and can recommend niche items that very few other users are interested in.

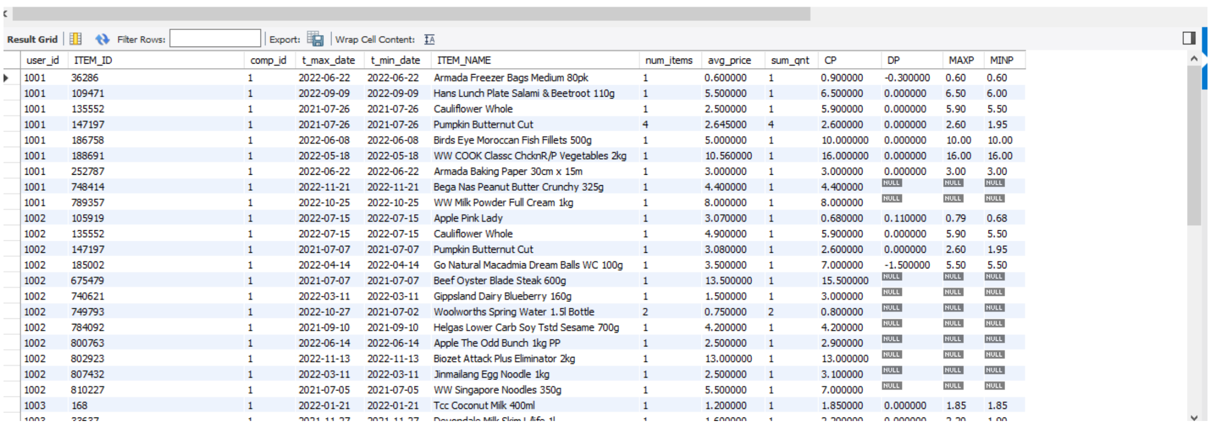
- **Main components of Content-based Filtering approach**

Recommenders mostly have 3 components:

- **Candidate Generations**: generating smaller subsets of candidates to recommend to a user, given a huge pool of thousands of items.
  For this step, I collected user data by fetching the database using the following query.

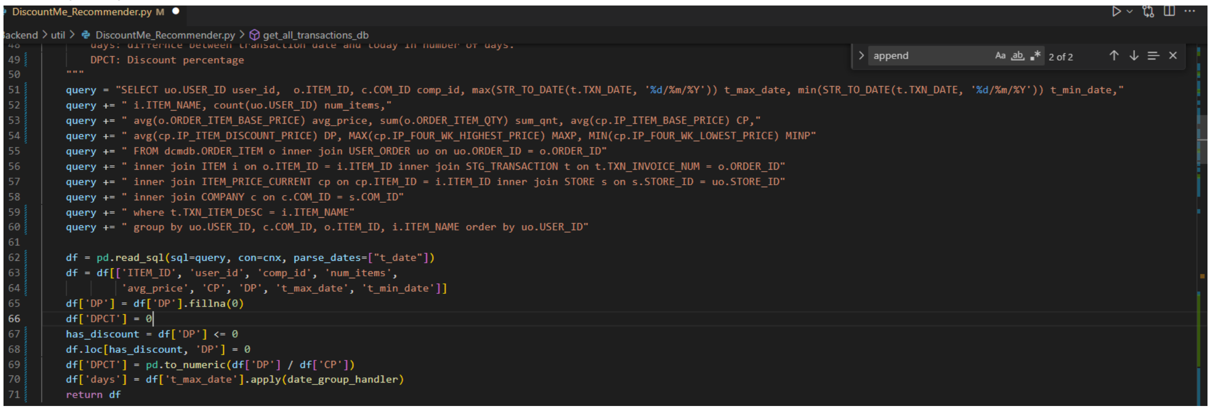


```sql
use dcmdb;

SELECT uo.USER_ID user_id,  o.ITEM_ID, c.COM_ID comp_id, max(STR_TO_DATE(t.TXN_DATE, '%d/%m/%Y')) t_max_date, min(STR_TO_DATE(t.TXN_DATE, '%d/%m/%Y')) t_min_date,
i.ITEM_NAME, count(uo.USER_ID) num_items, avg(o.ORDER_ITEM_BASE_PRICE) avg_price, sum(o.ORDER_ITEM_QTY) sum_qnt, avg(cp.IP_ITEM_BASE_PRICE) CP,
avg(cp.IP_ITEM_DISCOUNT_PRICE) DP, MAX(cp.IP_FOUR_WK_HIGHEST_PRICE) MAXP, MIN(cp.IP_FOUR_WK_LOWEST_PRICE) MINP
FROM dcmdb.ORDER_ITEM o inner join USER_ORDER uo on uo.ORDER_ID = o.ORDER_ID
inner join ITEM i on o.ITEM_ID = i.ITEM_ID inner join STG_TRANSACTION t on t.TXN_INVOICE_NUM = o.ORDER_ID
inner join ITEM_PRICE_CURRENT cp on cp.ITEM_ID = i.ITEM_ID inner join STORE s on s.STORE_ID = uo.STORE_ID
inner join COMPANY c on c.COM_ID = s.COM_ID
where t.TXN_ITEM_DESC = i.ITEM_NAME
group by uo.USER_ID, c.COM_ID, o.ITEM_ID, i.ITEM_NAME order by uo.USER_ID
```



user_id	ITEM_ID	comp_id	t_max_date	t_min_date	ITEM_NAME	num_items	avg_price	sum_qnt	CP	DP	MAXP	MINP
1001	36286	1	2022-06-22	2022-06-22	Armada Freezer Bags Medium 80pk	1	0.600000	1	0.900000	-0.300000	0.60	0.60
1001	109471	1	2022-09-09	2022-09-09	Hans Lunch Plate Salami & Beetroot 110g	1	5.500000	1	6.500000	0.000000	6.50	6.00
1001	135552	1	2021-07-26	2021-07-26	Cauliflower Whole	1	2.500000	1	5.900000	0.000000	5.90	5.50
1001	147197	1	2021-07-26	2021-07-26	Pumpkin Butternut Cut	4	2.645000	4	2.600000	0.000000	2.60	1.95
1001	186758	1	2022-06-08	2022-06-08	Birds Eye Moroccan Fish Fillets 500g	1	5.000000	1	10.000000	0.000000	10.00	10.00
1001	188691	1	2022-05-18	2022-05-18	WW COOK Classc ChcknR/P Vegetables 2kg	1	10.560000	1	16.000000	0.000000	16.00	16.00
1001	252787	1	2022-06-22	2022-06-22	Armada Baking Paper 30cm x 15m	1	3.000000	1	3.000000	0.000000	3.00	3.00
1001	748414	1	2022-11-21	2022-11-21	Bega Nas Peanut Butter Crunchy 325g	1	4.400000	1	4.400000	NULL	NULL	NULL
1001	789357	1	2022-10-25	2022-10-25	WW Milk Powder Full Cream 1kg	1	8.000000	1	8.000000	NULL	NULL	NULL
1002	105919	1	2022-07-15	2022-07-15	Apple Pink Lady	1	3.070000	1	0.680000	0.110000	0.79	0.68
1002	135552	1	2022-07-15	2022-07-15	Cauliflower Whole	1	4.900000	1	5.900000	0.000000	5.90	5.50
1002	147197	1	2021-07-07	2021-07-07	Pumpkin Butternut Cut	1	3.080000	1	2.600000	0.000000	2.60	1.95
1002	185002	1	2022-04-14	2022-04-14	Go Natural Macadmia Dream Balls WC 100g	1	3.500000	1	7.000000	-1.500000	5.50	5.50
1002	675479	1	2021-07-07	2021-07-07	Beef Oyster Blade Steak 600g	1	13.500000	1	15.500000	NULL	NULL	NULL
1002	740621	1	2022-03-11	2022-03-11	Gippsland Dairy Blueberry 160g	1	1.500000	1	3.000000	NULL	NULL	NULL
1002	749793	1	2022-10-27	2021-07-02	Woolworths Spring Water 1.5l Bottle	2	0.750000	2	0.800000	NULL	NULL	NULL
1002	784092	1	2021-09-10	2021-09-10	Helgas Lower Carb Soy Tstd Sesame 700g	1	4.200000	1	4.200000	NULL	NULL	NULL
1002	800763	1	2022-06-14	2022-06-14	Apple The Odd Bunch 1kg PP	1	2.500000	1	2.900000	NULL	NULL	NULL
1002	802923	1	2022-11-13	2022-11-13	Biozet Attack Plus Eliminator 2kg	1	13.000000	1	13.000000	NULL	NULL	NULL
1002	807432	1	2022-03-11	2022-03-11	Jinmailang Egg Noodle 1kg	1	2.500000	1	3.100000	NULL	NULL	NULL
1002	810227	1	2021-07-05	2021-07-05	WW Singapore Noodles 350g	1	5.500000	1	7.000000	NULL	NULL	NULL
1003	168	1	2022-01-21	2022-01-21	Tcc Coconut Milk 400ml	1	1.200000	1	1.850000	0.000000	1.85	1.85

This query used in code to user data.

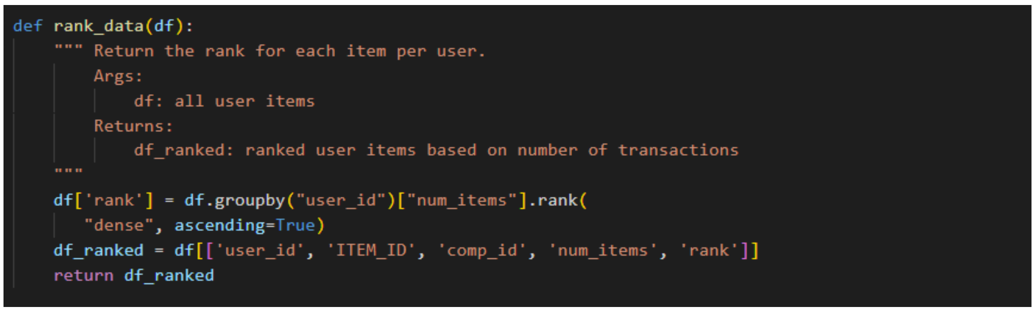


DiscountMe_Recommender.py M

ackend > util > DiscountMe_Recommender.py > get_all_transactions_db

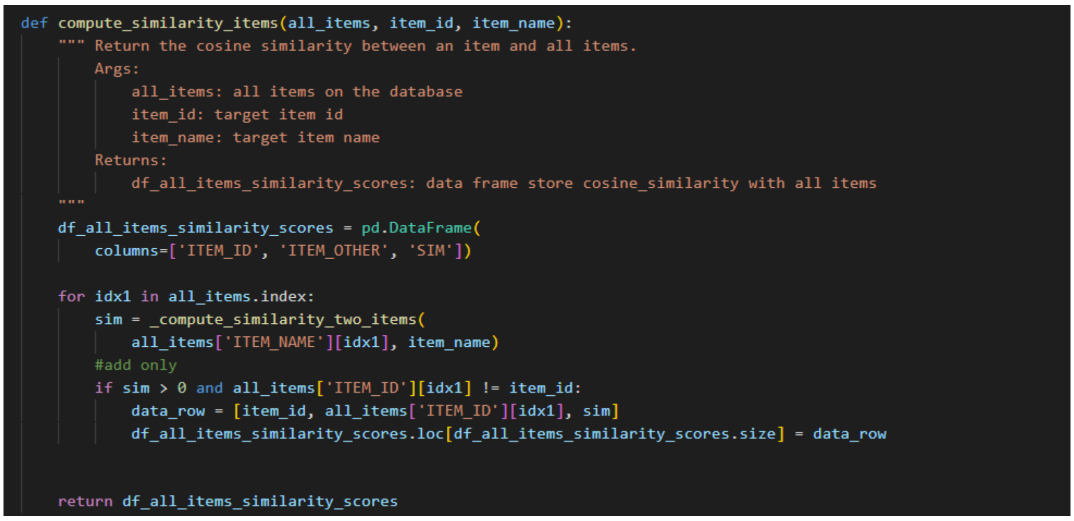
```python
        days: difference between transaction date and today in number of days.
        DPCT: Discount percentage
    """
    query = "SELECT uo.USER_ID user_id,  o.ITEM_ID, c.COM_ID comp_id, max(STR_TO_DATE(t.TXN_DATE, '%d/%m/%Y')) t_max_date, min(STR_TO_DATE(t.TXN_DATE, '%d/%m/%Y')) t_min_date,"
    query += " i.ITEM_NAME, count(uo.USER_ID) num_items,"
    query += " avg(o.ORDER_ITEM_BASE_PRICE) avg_price, sum(o.ORDER_ITEM_QTY) sum_qnt, avg(cp.IP_ITEM_BASE_PRICE) CP,"
    query += " avg(cp.IP_ITEM_DISCOUNT_PRICE) DP, MAX(cp.IP_FOUR_WK_HIGHEST_PRICE) MAXP, MIN(cp.IP_FOUR_WK_LOWEST_PRICE) MINP"
    query += " FROM dcmdb.ORDER_ITEM o inner join USER_ORDER uo on uo.ORDER_ID = o.ORDER_ID"
    query += " inner join ITEM i on o.ITEM_ID = i.ITEM_ID inner join STG_TRANSACTION t on t.TXN_INVOICE_NUM = o.ORDER_ID"
    query += " inner join ITEM_PRICE_CURRENT cp on cp.ITEM_ID = i.ITEM_ID inner join STORE s on s.STORE_ID = uo.STORE_ID"
    query += " inner join COMPANY c on c.COM_ID = s.COM_ID"
    query += " where t.TXN_ITEM_DESC = i.ITEM_NAME"
    query += " group by uo.USER_ID, c.COM_ID, o.ITEM_ID, i.ITEM_NAME order by uo.USER_ID"

    df = pd.read_sql(sql=query, con=cnx, parse_dates=["t_date"])
    df = df[['ITEM_ID', 'user_id', 'comp_id', 'num_items',
             'avg_price', 'CP', 'DP', 't_max_date', 't_min_date']]
    df['DP'] = df['DP'].fillna(0)
    df['DPCT'] = 0
    has_discount = df['DP'] <= 0
    df.loc[has_discount, 'DP'] = 0
    df['DPCT'] = pd.to_numeric(df['DP'] / df['CP'])
    df['days'] = df['t_max_date'].apply(date_group_handler)
    return df
```

- **Scoring Systems**: this step involves scoring to each of the items in the subsets. This is done by the Scoring system.
  One problem to build the scoring system is the unavailability of item rating data which should be in order_item table.
  To solve the problem, I proposed the following algorithm to compute the score for each item in respect to user behaviour:
  1. Rank user items (current ones) based on number of purchases (quantity).



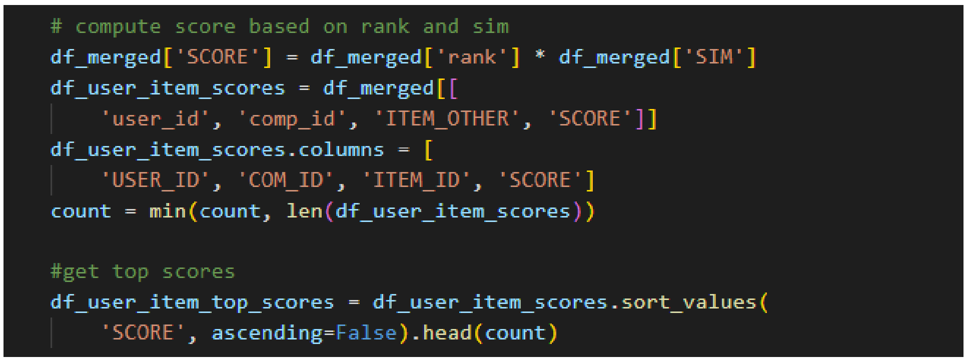
```
def rank_data(df):
    """ Return the rank for each item per user.
        Args:
            df: all user items
        Returns:
            df_ranked: ranked user items based on number of transactions
    """
    df['rank'] = df.groupby("user_id")["num_items"].rank(
        "dense", ascending=True)
    df_ranked = df[['user_id', 'ITEM_ID', 'comp_id', 'num_items', 'rank']]
    return df_ranked
```

  2. For each current item, compute the similarity with all other items and only conder similarity score > 0.



```
def compute_similarity_items(all_items, item_id, item_name):
    """ Return the cosine similarity between an item and all items.
        Args:
            all_items: all items on the database
            item_id: target item id
            item_name: target item name
        Returns:
            df_all_items_similarity_scores: data frame store cosine_similarity with all items
    """
    df_all_items_similarity_scores = pd.DataFrame(
        columns=['ITEM_ID', 'ITEM_OTHER', 'SIM'])

    for idx1 in all_items.index:
        sim = _compute_similarity_two_items(
            all_items['ITEM_NAME'][idx1], item_name)
        #add only
        if sim > 0 and all_items['ITEM_ID'][idx1] != item_id:
            data_row = [item_id, all_items['ITEM_ID'][idx1], sim]
            df_all_items_similarity_scores.loc[df_all_items_similarity_scores.size] = data_row


    return df_all_items_similarity_scores
```

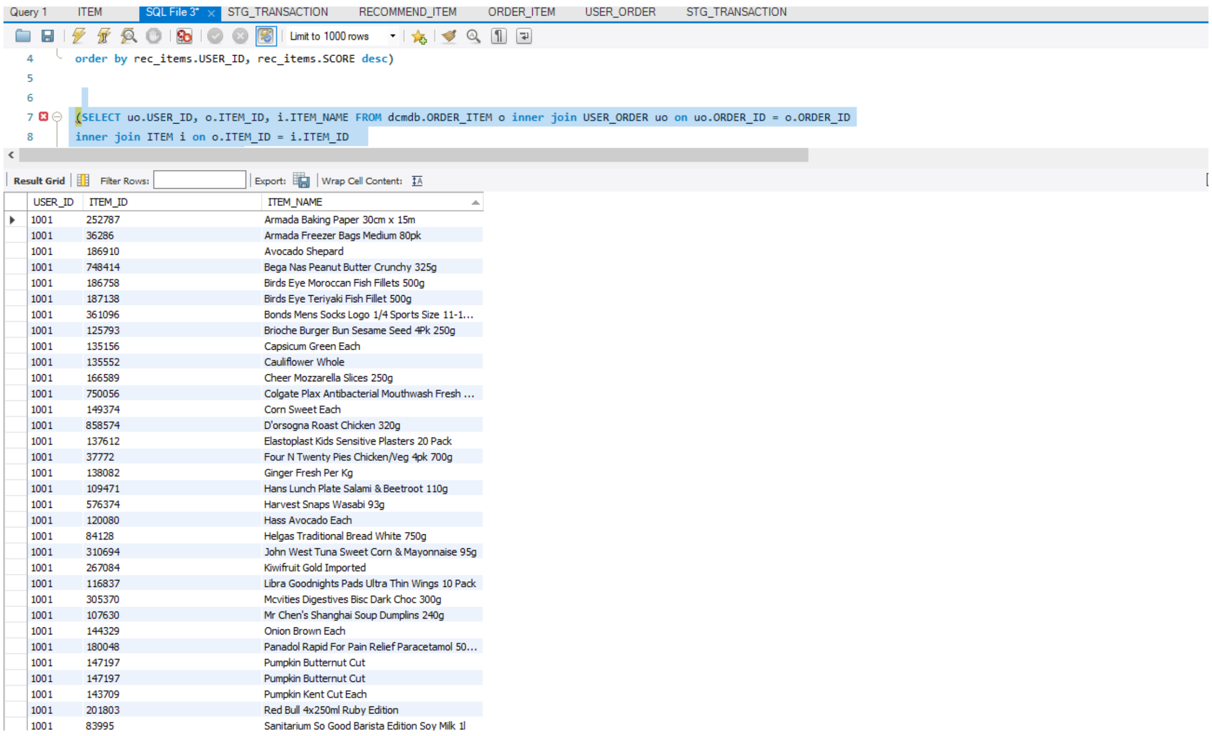
  3. Compute the score as a product of rank of related item and similarity score.



```
# compute score based on rank and sim
df_merged['SCORE'] = df_merged['rank'] * df_merged['SIM']
df_user_item_scores = df_merged[[
    'user_id', 'comp_id', 'ITEM_OTHER', 'SCORE']]
df_user_item_scores.columns = [
    'USER_ID', 'COM_ID', 'ITEM_ID', 'SCORE']
count = min(count, len(df_user_item_scores))

#get top scores
df_user_item_top_scores = df_user_item_scores.sort_values(
    'SCORE', ascending=False).head(count)
```

  4. Select items with top scores.

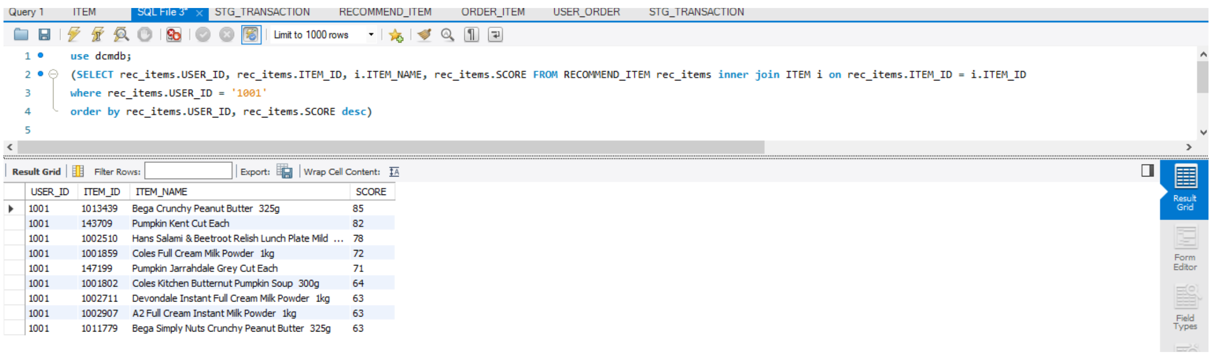
- **Sample user current items**



```
order by rec_items.USER_ID, rec_items.SCORE desc)


(SELECT uo.USER_ID, o.ITEM_ID, i.ITEM_NAME FROM dcmdb.ORDER_ITEM o inner join USER_ORDER uo on uo.ORDER_ID = o.ORDER_ID
inner join ITEM i on o.ITEM_ID = i.ITEM_ID
```

USER_ID	ITEM_ID	ITEM_NAME
1001	252787	Armada Baking Paper 30cm x 15m
1001	36286	Armada Freezer Bags Medium 80pk
1001	186910	Avocado Shepard
1001	748414	Bega Nas Peanut Butter Crunchy 325g
1001	186758	Birds Eye Moroccan Fish Fillets 500g
1001	187138	Birds Eye Teriyaki Fish Fillet 500g
1001	361096	Bonds Mens Socks Logo 1/4 Sports Size 11-1...
1001	125793	Brioche Burger Bun Sesame Seed 4Pk 250g
1001	135156	Capsicum Green Each
1001	135552	Cauliflower Whole
1001	166589	Cheer Mozzarella Slices 250g
1001	750056	Colgate Plax Antibacterial Mouthwash Fresh ...
1001	149374	Corn Sweet Each
1001	858574	D'orsogna Roast Chicken 320g
1001	137612	Elastoplast Kids Sensitive Plasters 20 Pack
1001	37772	Four N Twenty Pies Chicken/Veg 4pk 700g
1001	138082	Ginger Fresh Per Kg
1001	109471	Hans Lunch Plate Salami & Beetroot 110g
1001	576374	Harvest Snaps Wasabi 93g
1001	120080	Hass Avocado Each
1001	84128	Helgas Traditional Bread White 750g
1001	310694	John West Tuna Sweet Corn & Mayonnaise 95g
1001	267084	Kiwifruit Gold Imported
1001	116837	Libra Goodnights Pads Ultra Thin Wings 10 Pack
1001	305370	Mcvities Digestives Bisc Dark Choc 300g
1001	107630	Mr Chen's Shanghai Soup Dumplins 240g
1001	144329	Onion Brown Each
1001	180048	Panadol Rapid For Pain Relief Paracetamol 50...
1001	147197	Pumpkin Butternut Cut
1001	147197	Pumpkin Butternut Cut
1001	143709	Pumpkin Kent Cut Each
1001	201803	Red Bull 4x250ml Ruby Edition
1001	83995	Sanitarium So Good Barista Edition Soy Milk 1l

- **Recommended items for a user with scoring results.**



```
use dcmdb;
(SELECT rec_items.USER_ID, rec_items.ITEM_ID, i.ITEM_NAME, rec_items.SCORE FROM RECOMMEND_ITEM rec_items inner join ITEM i on rec_items.ITEM_ID = i.ITEM_ID
where rec_items.USER_ID = '1001'
order by rec_items.USER_ID, rec_items.SCORE desc)
```

USER_ID	ITEM_ID	ITEM_NAME	SCORE
1001	1013439	Bega Crunchy Peanut Butter 325g	85
1001	143709	Pumpkin Kent Cut Each	82
1001	1002510	Hans Salami & Beetroot Relish Lunch Plate Mild ...	78
1001	1001859	Coles Full Cream Milk Powder 1kg	72
1001	147199	Pumpkin Jarrahdale Grey Cut Each	71
1001	1001802	Coles Kitchen Butternut Pumpkin Soup 300g	64
1001	1002711	Devondale Instant Full Cream Milk Powder 1kg	63
1001	1002907	A2 Full Cream Instant Milk Powder 1kg	63
1001	1011779	Bega Simply Nuts Crunchy Peanut Butter 325g	63