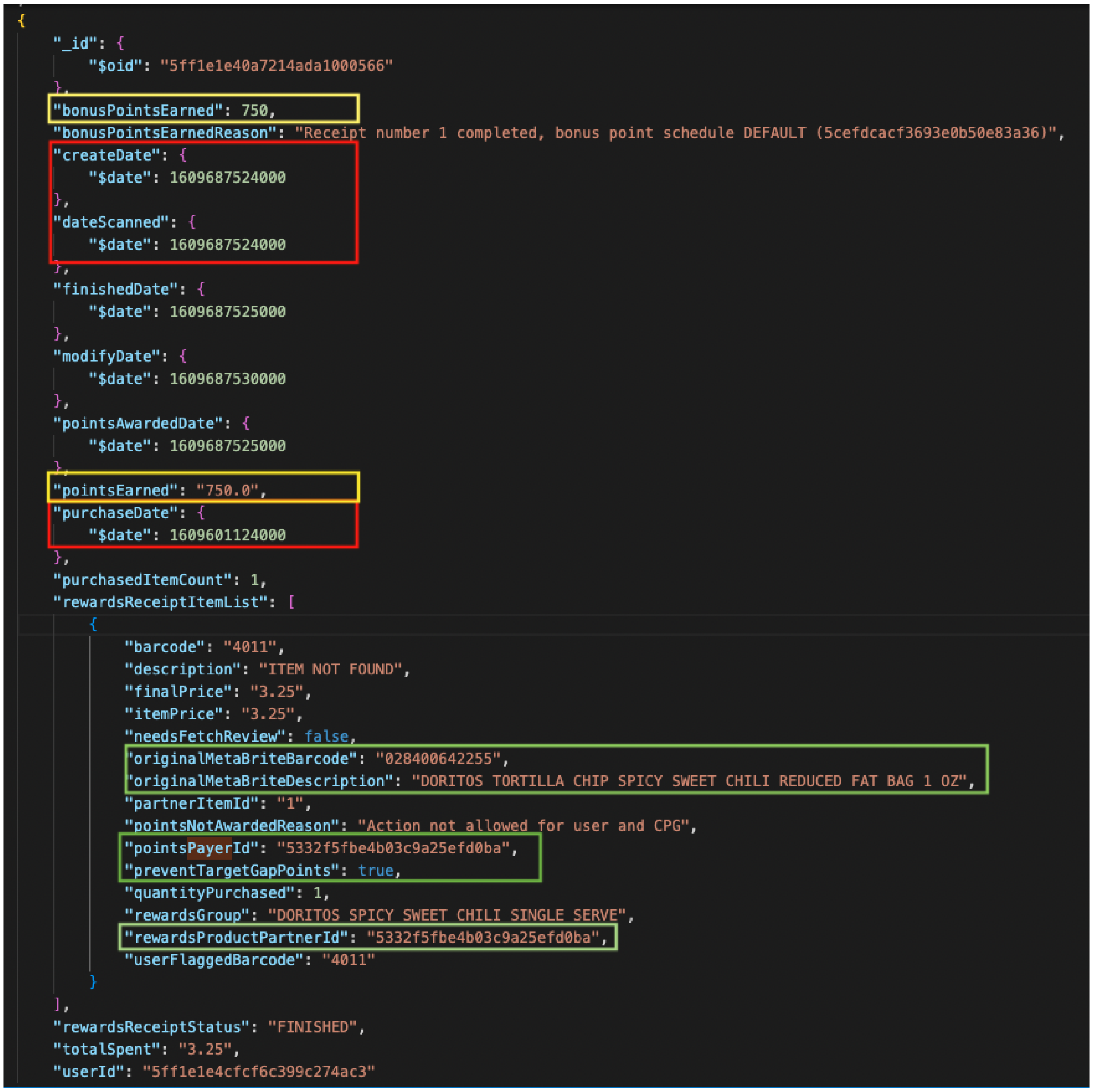
Hi team,

I’ve been working on analyzing our Receipts, Users and Brands data and have conducted some analysis. I’m sharing some of my open questions, data discrepancies, suggestions on additional data to track as well as recommendations on how to scale.

<u>Open questions</u>

Regarding the **Receipts** data, I’ve attached a screenshot of one receipt below. From here:

- Is createDate different to dateScanned or purchaseDate (highlighted in red below)? I imagine createDate is the date the item was purchased, and dateScanned is the date that it was scanned - so do we need a third field for purchaseDate?
- Is pointsEarned an aggregate of base points earned + bonusPointsearned? Or is the total points on a receipt = pointsEarned + bonusPointsEarned(highlighted yellow)
- Under the rewardsReceiptItemList, what do the following fields mean (highlighted green)-
    - Both the ‘metabrite’ fields
    - preventTargetGapPoints
    - Is pointsPayerID the same as rewardsProductpartnerID?

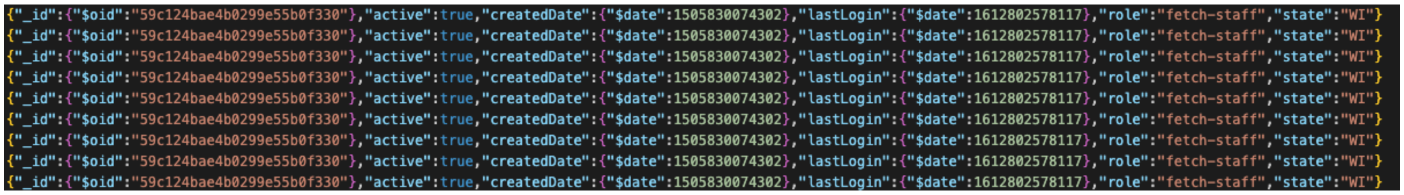


```
{
    "_id": {
        "$oid": "5ff1e1e40a7214ada1000566"
    },
    "bonusPointsEarned": 750,
    "bonusPointsEarnedReason": "Receipt number 1 completed, bonus point schedule DEFAULT (5cefdcacf3693e0b50e83a36)",
    "createDate": {
        "$date": 1609687524000
    },
    "dateScanned": {
        "$date": 1609687524000
    },
    "finishedDate": {
        "$date": 1609687525000
    },
    "modifyDate": {
        "$date": 1609687530000
    },
    "pointsAwardedDate": {
        "$date": 1609687525000
    },
    "pointsEarned": "750.0",
    "purchaseDate": {
        "$date": 1609601124000
    },
    "purchasedItemCount": 1,
    "rewardsReceiptItemList": [
        {
            "barcode": "4011",
            "description": "ITEM NOT FOUND",
            "finalPrice": "3.25",
            "itemPrice": "3.25",
            "needsFetchReview": false,
            "originalMetaBriteBarcode": "028400642255",
            "originalMetaBriteDescription": "DORITOS TORTILLA CHIP SPICY SWEET CHILI REDUCED FAT BAG 1 OZ",
            "partnerItemId": "1",
            "pointsNotAwardedReason": "Action not allowed for user and CPG",
            "pointsPayerId": "5332f5fbe4b03c9a25efd0ba",
            "preventTargetGapPoints": true,
            "quantityPurchased": 1,
            "rewardsGroup": "DORITOS SPICY SWEET CHILI SINGLE SERVE",
            "rewardsProductPartnerId": "5332f5fbe4b03c9a25efd0ba",
            "userFlaggedBarcode": "4011"
        }
    ],
    "rewardsReceiptStatus": "FINISHED",
    "totalSpent": "3.25",
    "userId": "5ff1e1e4cfcf6c399c274ac3"
```

Wondering if these questions have been asked in the past - if they're causing confusion, maybe we should think about renaming some of them to be clearer.

Data discrepancies

I found a lot of duplicates in the User data, like this user:



```
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
{"_id":{"$oid":"59c124bae4b0299e55b0f330"},"active":true,"createdDate":{"$date":1505830074302},"lastLogin":{"$date":1612802578117},"role":"fetch-staff","state":"WI"}
```

We should find these duplicates and remove them asap.

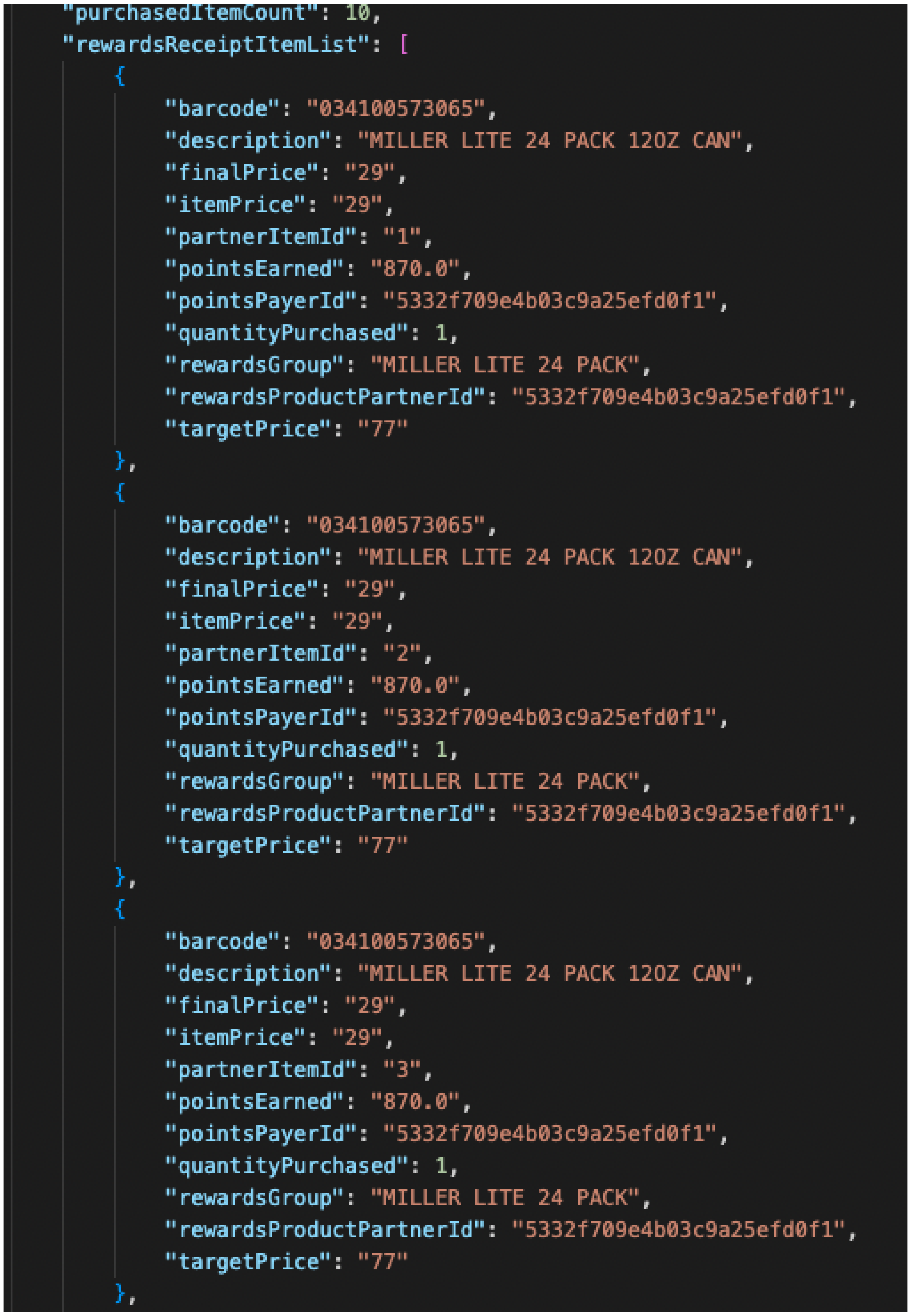
You can run this query to see the number of duplicate entries in the data:

```python
def num_unique_users(data):
    unique_users = []
    for entry in data:
        user_id = entry['_id']['$oid']
        if user_id not in unique_users:
            unique_users.append(user_id)

    print('total entries in database: ' + str(len(data)))
    print('total unique entries: ' + str(len(unique_users)))

num_unique_users(users) #users refers to the users.json file
```

I also found an issue with some of the receipts data- If you look at the screenshot below, you'll see that this receipt had multiple of the same item - same barcode, description, price etc. But the 'partnerItemId' is different. Is there a reason for that? And is there a reason we don't collapse all the repeat items into one, and populate the 'quantityPurchased' field accordingly?



```
    "purchasedItemCount": 10,
    "rewardsReceiptItemList": [
        {
            "barcode": "034100573065",
            "description": "MILLER LITE 24 PACK 12OZ CAN",
            "finalPrice": "29",
            "itemPrice": "29",
            "partnerItemId": "1",
            "pointsEarned": "870.0",
            "pointsPayerId": "5332f709e4b03c9a25efd0f1",
            "quantityPurchased": 1,
            "rewardsGroup": "MILLER LITE 24 PACK",
            "rewardsProductPartnerId": "5332f709e4b03c9a25efd0f1",
            "targetPrice": "77"
        },
        {
            "barcode": "034100573065",
            "description": "MILLER LITE 24 PACK 12OZ CAN",
            "finalPrice": "29",
            "itemPrice": "29",
            "partnerItemId": "2",
            "pointsEarned": "870.0",
            "pointsPayerId": "5332f709e4b03c9a25efd0f1",
            "quantityPurchased": 1,
            "rewardsGroup": "MILLER LITE 24 PACK",
            "rewardsProductPartnerId": "5332f709e4b03c9a25efd0f1",
            "targetPrice": "77"
        },
        {
            "barcode": "034100573065",
            "description": "MILLER LITE 24 PACK 12OZ CAN",
            "finalPrice": "29",
            "itemPrice": "29",
            "partnerItemId": "3",
            "pointsEarned": "870.0",
            "pointsPayerId": "5332f709e4b03c9a25efd0f1",
            "quantityPurchased": 1,
            "rewardsGroup": "MILLER LITE 24 PACK",
            "rewardsProductPartnerId": "5332f709e4b03c9a25efd0f1",
            "targetPrice": "77"
        },
```

Additional data we should track

The current database is an excellent start to capturing important data, but I have some suggestions for next steps:

1. User table:

    a. Record every user session (not just lastLogin time and createdDate)
    b. Determine user engagement with this data and learn about customer behavior
    c. Can lead to improved personalization and targeted push notifications
    d. E.g. if a user is most active on Mondays, Tuesdays and Thursdays at 6-9pm, we can send them push notifications on those days

2. Categorize our data further
    a. Come up with a category taxonomy system to maintain a quickly-growing items catalog
    b. Use this to further understand user behavior with regards to purchasing and getting rewards. Eg. home product category vs food and beverage category
    c. Will be useful in the ‘Special Offers → category’ filter in the app
3. Receipts:
    a. Currently missing source tracking where the purchase was made. E.g. Online vs in-person, big box retailer vs small business
    b. Flag receipts that come via in-app integrations (Amazon, Walmart, etc) since those can be monetized differently.
4. Tracking rewards more granularly
    a. Track item level vs brand level rewards separately. E.g. Gatorade zero protein gives me 100% back (item level reward) + 10 points for every dollar spend (brand level reward)
    b. The database has pointsEarned and bonusPointsEarned, but nothing about breaking up pointsEarned into item-points and brand-points.

Recommendations for scaling

Based on my analysis, I recommend that we migrate to a relational database model. I’ve attached my proposal for a high-level database schema that we can adopt. Some of the advantages of this approach are:

- Allows us to more clearly define our data models
    - Also allows for flexibility if schema needs to be updated
- Reduces chances of data duplication due to normalization
- Much easier to run analysis

As we continue to scale, there will be other concerns. Here are some ways we can mitigate these concerns:

- If the DB becomes too slow, add DB replication to improve read performance and be fault tolerant
    - Primary DB (read/write)
    - Secondary read only DBs
    - Add monitoring to check things like DB replication lag
- If not already, move DB, file storage and services to cloud infrastructure. Try to take advantage of auto-scaling.
- Identify common query patterns and/or bottlenecks and add caching to improve performance if needed.

- Use service oriented architecture to avoid single points of failures. Use tools like Docker and Kubernetes to deploy and manage service instances.
- Build dashboards that monitor our app latency which directly affects customer experience. Set time thresholds on important queries and have team members get alerted.

I hope my recommendations make sense, happy to discuss further. But the first action item is to clean up the current database.