ATAK Cosmetics

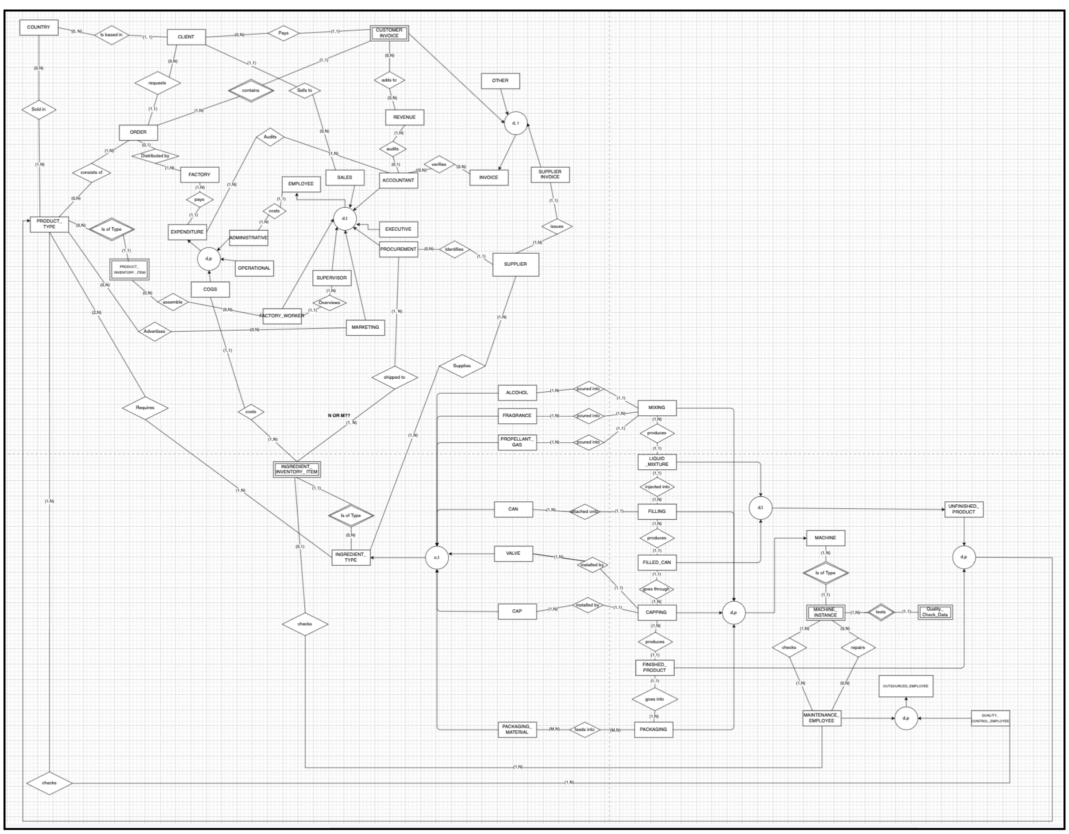
Team 7

Nikita Boddu, Kaan Gezguc, Noah Kaminer, Begum Dogan, Yash Bhandari, Carrie Liu, Erel Saul, Jillian Criscuolo, Majd Muna, Nicole Guzhavin, Jonathan Brian

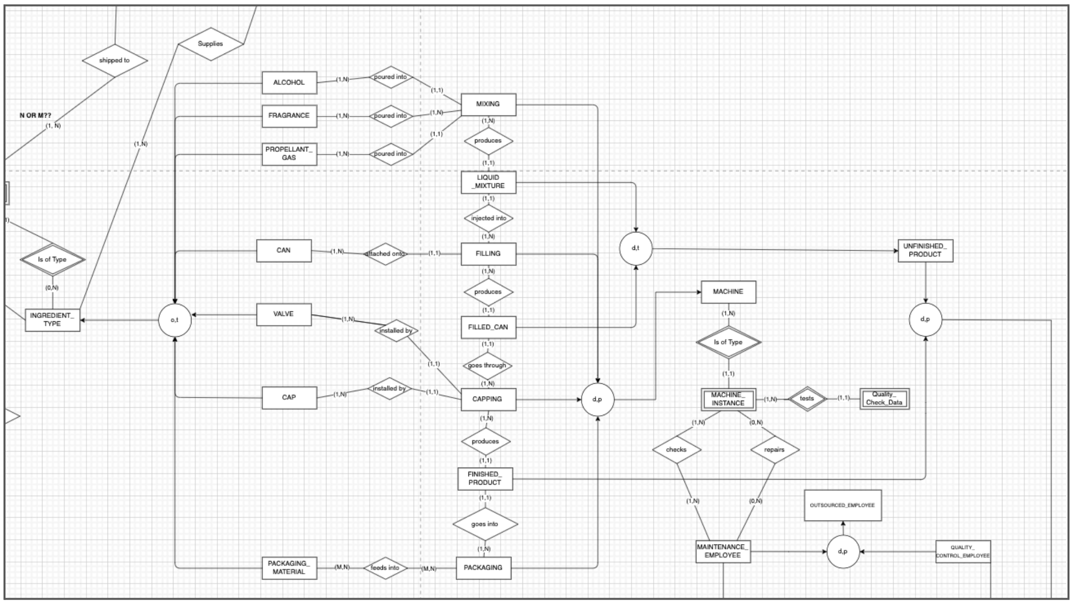
Executive Summary

ATAK Cosmetics is an international manufacturer and exporter of cosmetics based in Istanbul, Turkey specializing in Private Label Products from the beginning of the design and R&D phase and fragrance decision until the shipment of the final products. ATAK Cosmetics currently has around 230 employees and they have clients in over 50 countries. Since its inception, it has been one of the pioneer companies in the sector with its aerosol and liquid products and R&D works. The vast majority of the data ATAK keeps track of are the order agreements, pricing sheets, manufacturing and products details, and CRM information (Customer Relationship Management). We have updated ATAK's database infrastructure from Microsoft Excel files stored on a cloud data service to a MySQL database that tracks the entire manufacturing process from ingredient inventory, assembly of the product, and quality assurance checks all the way to employee and financial records that exist within the factory.

Simplified EER:



- Added focus on the stages of manufacturing, tracking the ingredients and machines all the way to the final product

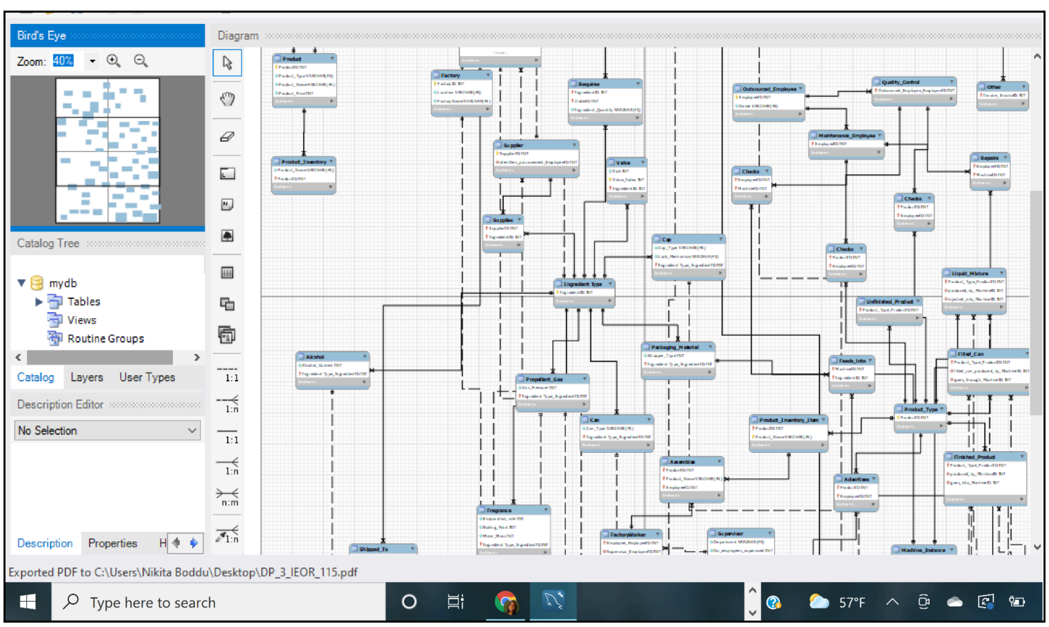


Relational Design (Schema):

1. Country(Country_Name, Region)
2. Client(ClientID, Name, First_Order, First_Order_Date, sells_to_Sales_EmployeeID[6a], based_in_Country_Name[1])
3. Order(OrderID, Total_Price, Payment_Terms, Transportation, Date, Num_Units distributed_by_FactoryID[4], requested_by_ClientID[2])
4. Factory(FactoryID, Factory_Name, Location)
5. Expenditure(ExpenditureID, payment_to_factory[4])
    a. Administrative(ExpenditureID[6])
    b. Operational(ExpenditureID[6])
6. Employee(EmployeeID, administrative_cost_EmployeeID[5a])
    a. Sales(EmployeeID[6], Total_sales)
    b. Accountant(EmployeeID[6], audited_revenue_timestamp[7])
    c. Procurement(EmployeeID[6], No_supplier_contacts)
    d. Marketing(EmployeeID[6], Total_campaigns)
    e. Supervisor(EmployeeID[6], Department, No_employees_supervised)
    f. Factory_Worker(EmployeeID[6], overview_supervisor_EmployeeID[6f])
7. Revenue(Timestamp, Amount)
8. Invoice(InvoiceID, Amount, Due_Date, Billing_Date)
    a. Supplier_Invoice(InvoiceID[8], issued_supplier_SupplierID[9])
    b. Other(InvoiceID[8])
9. Supplier(SupplierID, identifies_procurement_EmployeeID[6])
10. Outsourced_Employee(EmployeeID, name)
    a. Maintenance_Employee(EmployeeID[10])
    b. Quality_Control_Employee(EmployeeID[10])
11. Machine(Machine_Name)
    a. Mixing(Machine_Name[11], poured_into_Ingredient_Name_1[13a], poured_into_Ingredient_Name_2[13c])
    b. Capping(Machine_Name[11], installed_by_Ingredient_Name_1[13e], installed_by_Ingredient_Name_2[13f])
    c. Filling(Machine_Name[11], attached_onto_Ingredient_Name[14d])
    d. Packaging(Machine_Name[11])
12. Product_Type(Product_Name, Product_Price)
    a. Unfinished_Product(Product_Name[12])
        i. Liquid_Mixure(Product_Name[12], produced_by_Machine_Name[11a], injected_into_Machine_Name[11c])
        ii. Filled_Can(Product_Name[12], goes_through_Machine_Name[11b], filled_can_produced_by_Machine_Name[11c])
    b. Finished_Product(Product_Name[12], produced_by_Machine_Name[11a], goes_into_Machine_Name[11d])

13. Ingredient_Type(Ingredient_Name)
    a. Alcohol(Ingredient_Name[13], Alcohol_content)
    b. Fragrance(Ingredient_Name[13], Evaporation_rate, Boiling_point, Molar_mass)
    c. Propellant_Gas(Ingredient_Name[13], Gas_pressure)
    d. Can(Ingredient_Name[13], Can_type)
    e. Valve(Ingredient_Name[13], Valve_index, Size)
    f. Cap(Ingredient_Name[13], Cap_type, Lock_mechanism)
    g. Packaging_Material(Ingredient_Name[13], Wrapper_type)
14. Customer_Invoice(InvoiceID[8], Email, Date, paid_by_ClientID[2], contains_OID[3])
15. Product_Inventory_Item(Product_Name[12], Product_Inventory_Number)
16. Ingredient_Inventory_Item(Ingredient_Name[13], Inventory_Number, Quantity)
17. Machine_instance(MachineID, Machine_name[11])
18. Quality_Check(MachineID[17], Machine_name[11], FailureID , Datetime, Fail_Type, Time_Since_Last_Failure)
19. Sold_in(Country_Name[1], ProductID[12])
20. Consists_of(OrderID[3], ProductID[12b])
21. Requires(ProductID[12a], Ingredient_Name[13], Ingredient_Quantity)
22. Advertises(ProductID[12], EmployeeID[6])
23. Assembles(ProductID[12], Product_name[15], EmployeeID[6])
24. Checks(ProductID[12], EmployeeID[10b])
25. Adds_to(Timestamp[7], OrderID[3], Email[14])
26. Verifies(EmployeeID[6b], InvoiceID[8])
27. Supplies(Ingredient_Name[13], SupplierID[9])
28. Poured_into(Ingredient_Name[13b], Machine_Name[11a])
29. Feeds_into(Ingredient_Name[13g], Machine_Name[11d])
30. Checks(MachineID[17], Machine_Name[11], EmployeeID[10a])
31. Repairs(MachineID[17], Machine_Name[11], EmployeeID[10a])
32. Client_email(CustomerID[2], e_addresses)
33. Order_details(OrderID[3], ProductIDs)

Tables in MySQL Relationship View



<u>Two Interesting Queries</u>

With great amounts of data and information neatly stored in the database, ATAK can leverage SQL queries to model predictive algorithms and probability distributions that will benefit the business' operations. By extracting the information within the queries, ATAK can use external software and programming languages such as Python to predict monthly units sold or machine failure time for instance. Here, we used the Python package, [Faker] to generate realistic "dummy" data to present two example queries:

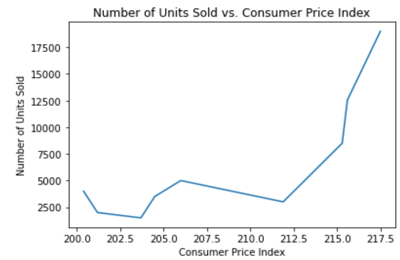
- Query 1: Extracting the units sold (end of month), number of orders, number of new clients, and total revenue for each month of each year

```
SELECT c.First_Order_Date,
       COUNT(o.OrderID) as Num_Orders,
       SUM(o.Total_Price) as Total_Revenue,
       SUM(o.Num_Units) as Num_Units_Sold,
       c.Num_New_Clients
FROM `Order` o, (SELECT COUNT(*) as Num_New_Clients,
                 DATE_FORMAT(First_Order, '%m%y') as First_Order_Date
              FROM `Client`
              GROUP BY First_Order_Date
              HAVING First_Order_date IS NOT NULL) as c
WHERE DATE_FORMAT(o.Date, '%m%y') = c.First_Order_Date
GROUP BY c.First_Order_Date
ORDER BY c.First_Order_Date;
```

First_Order_Date	Num_Orders	Total_Revenue	Num_Units_Sold	Num_New_Clients
0220	1	8387	3500	1
0320	1	6144	4000	1
0420	1	4215	2000	1
0520	3	12501	12550	1
0521	2	9446	3000	2
0620	1	5162	1500	1
0621	2	14598	19000	1
0921	1	7322	5000	1
1120	1	8223	NULL	1
1220	2	12052	8500	2

With this information, ATAK can forecast sales projections for efficient capital allocation. With that, they can better manage the inventory and can launch pinpointed marketing efforts to boost sales in particular months. For instance, if we augment the CPI data, the data can immediately tell us the relationship between monthly sales and average CPI.

	First_Order_Date	Num_Orders	Total_Revenue	Num_Units_Sold	Num_New_Clients	Month	Year	CPI	Month-Year
0	220	1	8387	3500.0	1	Feb	2020	204.5	Feb-2020
1	320	1	6144	4000.0	1	Mar	2020	200.4	Mar-2020
2	420	1	4215	2000.0	1	Apr	2020	201.2	Apr-2020
3	520	3	12501	12550.0	1	May	2020	215.6	May-2020
4	521	2	9446	3000.0	2	May	2021	211.9	May-2021
5	620	1	5162	1500.0	1	Jun	2020	203.7	June-2020
6	621	2	14598	19000.0	1	Jun	2021	217.5	June-2021
7	921	1	7322	5000.0	1	Sep	2021	206.0	Sep-2021
9	1220	2	12052	8500.0	2	Dec	2020	215.3	Dec-2020



Now, for modeling, we can predict the units sold using Linear Regression in the statsmodel package by using the independent variables: Revenue, # of orders, # of new clients, CPI, month, and year.

We can predict the number of units sold in a given month and year based on the Linear Regression equation below. For example, we are in December 2021, have 2 clients, 1 order, total revenue of 14562, and the CPI is 213.5, the number of units sold is predicted to be 4972 as shown below.

```
import statsmodels.formula.api as smf

ols1 = smf.ols(formula='Num_Units_Sold ~ Num_Orders + Num_New_Clients + Month + Total_R
               data=df)
model = ols1.fit()
print(model.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:          Num_Units_Sold   R-squared:                       1.000
Model:                             OLS   Adj. R-squared:                    nan
Method:                  Least Squares   F-statistic:                       nan
Date:                 Thu, 09 Dec 2021   Prob (F-statistic):                nan
Time:                         12:14:03   Log-Likelihood:                 175.69
No. Observations:                    9   AIC:                            -333.4
Df Residuals:                        0   BIC:                            -331.6
Df Model:                            8
Covariance Type:             nonrobust
===================================================================================
                      coef    std err          t      P>|t|      [0.025      0.975]
-----------------------------------------------------------------------------------
Intercept          10.1864        inf          0        nan        nan         nan
Month[T.Dec]    -1806.7741        inf         -0        nan        nan         nan
Month[T.Feb]    -5253.0998        inf         -0        nan        nan         nan
Month[T.Jun]    -2028.6966        inf         -0        nan        nan         nan
Month[T.Mar]    -1127.9785        inf         -0        nan        nan         nan
Month[T.May]    -4503.2354        inf         -0        nan        nan         nan
Month[T.Sep]    -3435.1347        inf         -0        nan        nan         nan
Year[T.2021]     1411.5204        inf          0        nan        nan         nan
Num_Orders        829.8375        inf          0        nan        nan         nan
Num_New_Clients -5189.7081        inf         -0        nan        nan         nan
Total_Revenue       1.6206        inf          0        nan        nan         nan
CPI                -2.3906        inf         -0        nan        nan         nan
```

The model tells us:

Num_Units_Sold = 10.19 - 5190(Num_New_Clients) + 830(Num_Orders) + 1.62(Total_Revenue) + -2.39(CPI) + -1807(T.Dec) - 5253(T.Feb) - 2029(T.June) - 1128(T.Mar) - 4503(T.May) - 3435(T.Sep) + 1412(T.2021)

```
int(model.predict(pd.DataFrame({"Num_New_Clients": [2],"Month":['Dec'],"Year": ['2021'],"CPI": [213.5],"Total_Revenue":[9514],"Nu
```

4972

Real data from ATAK was attempted to be used for this query, but ATAK unfortunately could not compile the needed data in due time. However, our ATAK client stated that the artificial data is a close representation of the real data and was satisfied with the model.

- Query 2: Fail type, average time between failures, and frequency of fail type for each machine token

```sql
SELECT q.Machine_Instance_MachineID, q.Machine_Instance_Machine_name, q.Fail_Type,
       COUNT(q.Machine_Instance_MachineID) as Frequency_of_Failure_Type,
       AVG(q.Time_Since_Last_Failure) as Average_Time_Between_Failures
FROM `Quality Check` q
GROUP BY q.Machine_Instance_MachineID, q.Machine_Instance_Machine_name, q.Fail_Type;
```

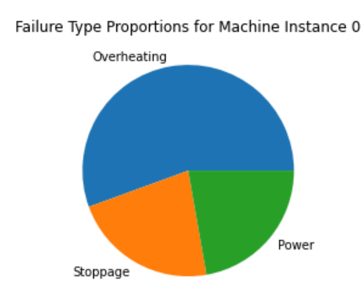
Machine_Instance_Ma...	Machine_Instance_Machine_na...	Fail_Type	Frequency_of_Failure_Ty...	Average_Time_Between_Failur...
0	Filling	Overheating	5	67.6
0	Filling	Stoppage	2	99
0	Filling	Power	2	77
1	Capping	Power	1	281
1	Capping	Stoppage	3	143
2	Packaging	Overheating	2	17
2	Packaging	Power	3	166.33333333333334
3	Mixing	Overheating	1	11
3	Mixing	Stoppage	1	0

With this information, ATAK can understand what type of problems are causing machine failures most frequently, estimate the time between failures, and correspondingly improve operational efficiency and maintenance scheduling. For instance, by grouping each machine instance together, we can analyze the failure breakdown:

In [10]:
```python
df.groupby(['Machine_Instance_MachineID']).sum()
```

Out[10]:

Machine_Instance_MachineID	Frequency_of_Failure_Type	Average_Time_Between_Failures
0	9	243.600000
1	4	424.000000
2	5	183.333333
3	2	11.000000



Furthermore, we can model failure times as a poisson process to estimate the likelihood of a failure at a certain time. For example, Machine_Instance_MachineID 0 frequently overheats, showing 5 occurrences with an average time of 67 days between failures. We use stochasticity and a poisson process to model the probability distribution for days between an overheating failure for this machine. We notice that most failures will likely be between 57 and 77 days apart (flexion point).

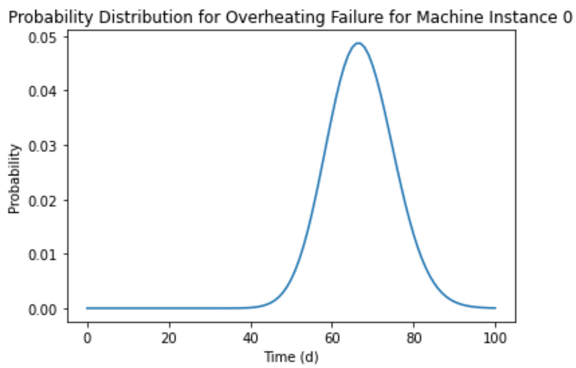
```python
from scipy.stats import poisson

storage =[]

for i in np.linspace(0,100,101):
    storage = np.append(storage,poisson.pmf(i,67))

storage;
```

```python
plt.plot(np.linspace(0,100,101), storage)
plt.title('Probability Distribution for Overheating Failure for Machine Instance 0')
plt.xlabel('Time (d)')
plt.ylabel('Probability');
```



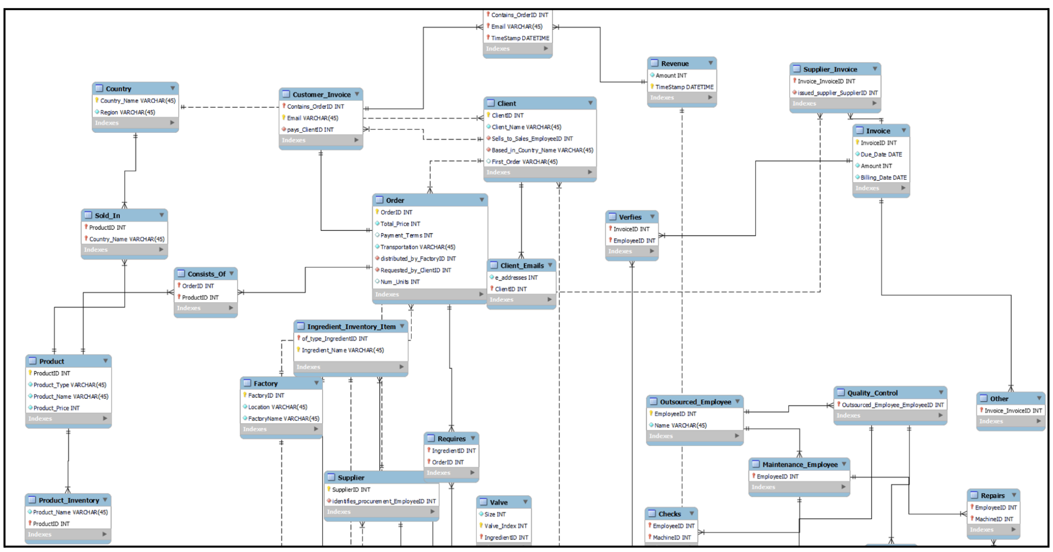
Final DP Report

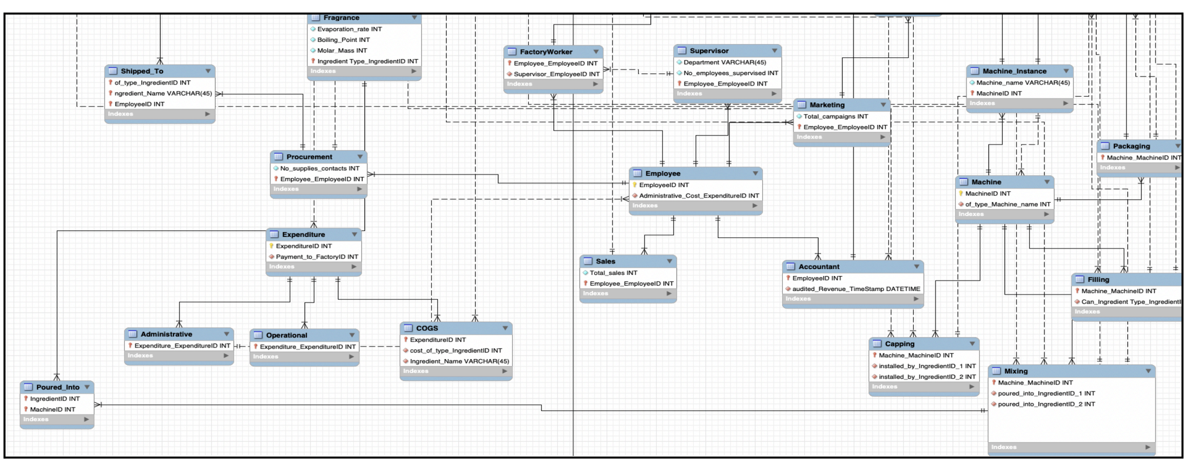
Client, Previous Approaches, and Goals

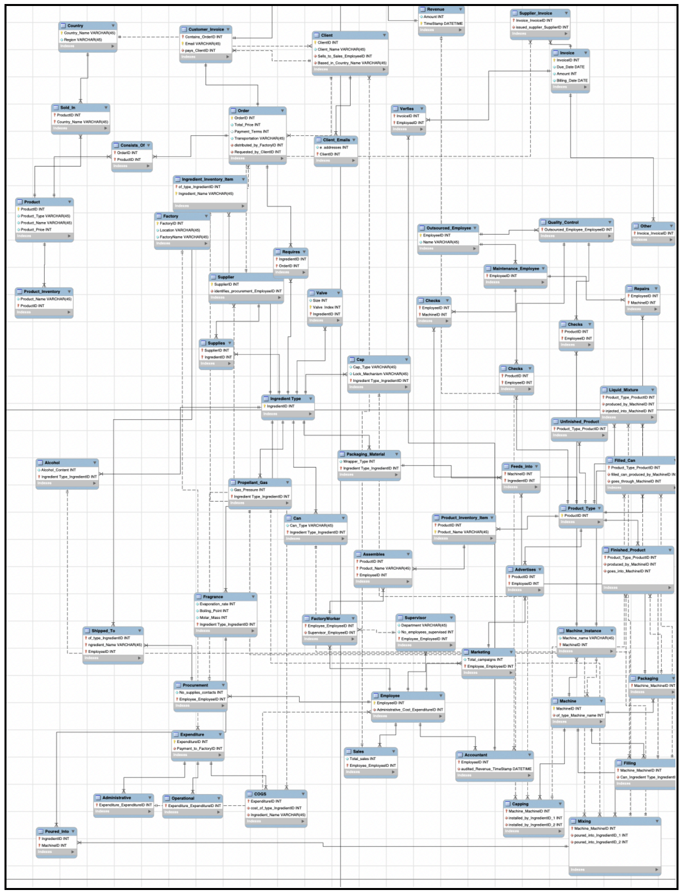
As a private international manufacturer and exporter of cosmetics, ATAK Cosmetics specializes in Private Label Products. Also known as “House of Private Label”, customers around the world use ATAK’s manufacturing facilities to bring their brands into life. ATAK Cosmetics manufactures a variety of different products, but their four main categories are: aerosol products (body spray, hair spray, air freshener), fragrance (perfume, body splash, cologne), liquid filling (shampoo, body lotion, shower gel), and home fragrance products (reed diffuser, electrical plug-in, decorative air freshener, gel air freshener). Among these categories, aerosol products constitute more than 50% of total production. Due to the varying product requests from different customers, ATAK’s manufacturing facility has to accommodate a large number of machinery to support different processes. The make-to-order nature of ATAK’s business requires them to simultaneously handle different customer orders and keep track of the production cycle of various products, from receiving the order from the customer to product design, to manufacturing, to quality assurance, and finally, to fulfilling the shipment.

From our initial conversations with ATAK we realized that they do not have a formal database system that is well organized and easy to navigate. Prior to our involvement, they recorded their data on multiple Excel spreadsheets and primarily captured limited information about their order sales, employees and inventory sourcing and stockpiles. Given their current state, we were tasked to help develop a consolidated and well-organized database on MySQL that captures more expanded information about ATAK's operations. Through our discussions with Mr. Saul and mentorship from Prof. Goldberg, we aimed to incorporate the manufacturing process into the database. This would enable ATAK to better monitor their machines, employee utilization and quality check protocols. Upon completing the aforementioned goals, we aimed to help ATAK improve their monitoring, but more importantly, create insightful queries to improve their operations and revenue generation analytics. For example, we set out to understand the frequency of machine failures and their root causes. This would enable ATAK to improve their machine maintenance schedules, reduce machine downtime and improve operational efficiency.

MySQL Screenshots Cont.







Normalization

In order to minimize data redundancy and eliminate unnecessary relationships, we identified four relations that require normalization. Here, we normalized them to 1NF, 2NF, 3NF, and BCNF forms sequentially. 1NF relations have single-valued attributes, and all of their domains require atomic elements. 2NF relations disallow partial dependencies, while 3NF relations disallow transitive dependencies. Finally, relations converted into BCNF satisfy the condition that for all functional dependencies {X -> Y}, X is a superkey. Here, the four relations, their FDs, and normalization forms are described as follows:

1. **Quality_Check** (<u>MachineID, Machine_Name, FailureID</u>, Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Machine_Start_Date, Components_Fixed)

Functional Dependencies F = {

A: {MachineID, MachineName, FailureID} → {Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Components_Fixed}

B: {Datetime} → {Weekday}

C: {MachineID, Machine_Name} → {Machine_Start_Date}

D: {Machine_Start_Date} → {Machine_Name}

}

(1NF): deals with multivalued attribute *'Components_Fixed'*
Quality_Check_Machine (MachineID, Machine_name, FailureID , Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Machine_Start_Date)
Components_Fixed (MachineID, Machine_name, FailureID, Component_Fixed)

(2NF): deals with partial dependencies existing in FD *C*
Quality_Check_Machine_Fix (MachineID, Machine_name, FailureID , Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Component_Fixed)
Machine_Lifetime (MachineID, Machine_name, Machine_Start_Date)
Components_Fixed (MachineID, Machine_name, FailureID, Component_Fixed)

(3NF): deals with transitive dependencies existing in FD *A*
Quality_Check_Machine (MachineID, Machine_name, FailureID , Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Machine_Start_Date)
Date_Day(Datetime, Weekday)
Machine_Lifetime (MachineID, Machine_name, Machine_Start_Date)
Components_Fixed (MachineID, Machine_name, FailureID, Component_Fixed)

(BCNF): deals with FD *F* which has Machine_name as a prime attribute
Quality_Check_Machine (MachineID, Machine_name, FailureID , Datetime, Weekday, Fail_Type, Time_Since_Last_Failure, Machine_Start_Date)
Date_Day(Datetime, Weekday)
Components_Fixed (MachineID, Machine_name, FailureID, Component_Fixed)
Start_Machines (Machine_Start_Date, Machine_name)
Machine_Item (Machine_name, MachineID)

2. **Customer_Invoice** (InvoiceID, Email, Date, ClientID, OrderID)

Functional Dependencies F = {

A: {InvoiceID, Email} -> {Date, OrderID}

B: {Email} -> {ClientID}

}

(1NF): Relation is already in 1NF because there are no multivalued attributes but is not in 2NF because not all non-prime attributes solely depend on the whole key

(2NF): deals with partial dependencies existing in FD *B*

Invoice_Instance (InvoiceID, Email, Date, OrderID)

Client_Info (Email, ClientID)

Already in 3NF and BCNF because there exists no transitive dependencies and for all FDs X -> Y, X is a superkey.

3. **Ingredient_Inventory_Item** (Ingredient_Name, Inventory_Number, Quantity, Max_Inventory, Eligible_Storage_Spaces)

Functional Dependencies F = {

A: {Ingredient_Name} -> {Max_Inventory, Eligible_Storage_Space}

B: {Ingredient_Name, Inventory_Number} -> {Quantity}

}

(1NF): deals with multivalued attribute *'Eligible_Storage_Spaces'*
Ingredient_Info (Ingredient_Name, Inventory_Number, Quantity, Max_Inventory)
Ingredient_Storage_Info (Ingredient_Name, Eligible_Storage_Space)

(2NF): deals with partial dependencies existing in FD *A*
Ingredient_Inventory_Info (Ingredient_Name, Max_Inventory)
Ingredient_Storage_Info (Ingredient_Name, Eligible_Storage_Space)
Ingredient_Instance (Ingredient_Name, Inventory_Number, Quantity)

Already in 3NF and BCNF because there exists no transitive dependencies and for all FDs X -> Y, X is a superkey.

4. **Client** (ClientID, EmployeeID, Client_Name, Country_Name, Date)

Functional Dependencies F = {

A: {ClientID} -> {Client_Name, Country_Name}

}

(1NF): Relation is already in 1NF because there are no multivalued attributes but is not in 2NF because not all non-prime attributes solely depend on the whole key

(2NF): deals with partial dependencies existing in FD *A*. Also, we record each Client - Employee interaction, and each client can interact with a specific employee more than once.

Client_Detail (ClientID, Client_Name, Country_Name)

Sales_Employee_Interaction (ClientID, EmployeeID, Date)

Already in 3NF and BCNF because there exists no transitive dependencies and for all FDs X -> Y, X is a superkey.

5. Product_Assembly (Product_name, Product_Inventory_Number, Ingredient_names, Employee_ID, Employee_Name, Machine_IDs)

Functional Dependencies F = {

A: {Product_name, Product_Inventory_Number} -> {Employee_ID}

B: {Product_name} → {Ingredients_List, Machine_IDs}

C: {Employee_ID} → {Employee_Name}

}

(1NF and 2NF): Relation is not in 1NF because of the multi-valued attribute "Inventory_Number" and "MachineIDs." These multi-valued attributes also cause the partial dependencies in FD *B*

Product_Assembly (Product_name, Product_Inventory_Number, Employee_ID, Employee_Name)
Required_Ingredient (Product_name, Ingredient_name)
Required_Machine (Product_name, Machine_ID)

(3NF): Relation is not in 2NF because of the transitive dependency in FD *C*
Product_Assembly (Product_name, Product_Inventory_Number, Employee_ID)
Required_Ingredient (Product_name, Ingredient_name)
Required_Machine (Product_name, Machine_ID)
Employee_Info (Employee_ID, Employee_Name)

Already in BCNF because for all FDs X -> Y, X is a superkey.

Team Contributions

- **Yash Bhandari, CCO**: Attended most weekly meetings, helped design the EER diagram (with a specific focus on the manufacturing process), helped design queries and write SQL code (especially for query 2) and lastly helped with ad hoc tasks like DP2 and final presentation slides etc.
- **Nikita Boddu**: Attended weekly meetings, EER diagram (creating entities and relationships, drew base of EER diagram on hand and in draw.io), brainstormed interesting queries, created EER diagram in SQL, forward engineered SQL model to create a medium we can code the queries on

- **Jonathan Brian** Assisted in revising EER diagram (entities, cardinality constraints), created presentation slides for DPII, performed normalization analysis from examining candidate relations to fully normalizing relations to 3NF/BCNF for DPIII and final report, created normalization analysis slides for DPIII, revised relational schema for final presentation.
- **Jillian Criscuolo**: Helped schedule weekly team meetings to address upcoming deadlines, EER diagram (organization and cardinality constraints), relational schema for DPII, wrote executive summary for DPIII, assisted with importing data from mySQL to Python and modifying the queries, worked on slide deck for Final DP Presentation, and worked to compile Final DP Report.
- **Begum Dogan**: Attended weekly meetings working with the group for DP reviews and revising EER diagram (entities and relationships), worked on the relational schema, worked on going over the final presentation and DP reviews and created their layout to be presented in the best way.
- **Kaan Gezguc, CEO**: Attended weekly meetings and helped design and revise the EER Diagram, revise the relational schema, keep the team on track with DP review and other deliverable deadlines, create layout and design slides for DP review and final presentations.
- **Nicole Guzhavin**: Helped revise EER diagram (entities and relationships), revised relational schema for DPII, performed normalization analysis from creating and picking relations to normalizing through 1NF, 2NF, 3NF, and BCNF for all, presented entire normalization analysis for the final project presentation
- **Noah Kaminer**: Helped keep project team on track with deadlines and progression of the project, helped create EER diagram, specifically focusing on the manufacturing process, devised interesting queries and statistical methods to analyze the data, helped create MySQL database from EER diagram, inputted data into MySQL, and ran necessary queries for analysis
- **Carrie Liu**: Helped devise EER diagram, working and editing the relational schema as we modified our diagram throughout the project, decking for DP reviews, performed normalization analysis for DPIII, final presentation, and final report.
- **Majd Muna**: Helped devise EER diagram (entities and relationships), created queries, coded and visualized the models from the query outputs, assisted with normalization analysis, presented during Final DP Presentation.
- **Erel Saul, Client Liaison**: Attended weekly meetings, maintained consistent communication between the client and the team to ensure client’s need were met, helped revise EER diagram (entities and relationships), lead the team in presenting during the DP presentations

Discussion and Future Work

Through our design project we were able to organize and prototype a centralized database for all of ATAK's data. By collecting and utilizing the necessary information, ATAK will be able to project sales to better allocate capital towards ingredient inventory and stock piles. With this, ATAK will be able to better manage its inventory and will even have the opportunity to launch targeted marketing efforts, which should boost yearly sales. For example, if we bring in CPI data, we can immediately determine the relationship between monthly sales and average CPI. In Query 1, we show how ATAK could use Linear Regression to predict the units sold through the following independent variables: revenue, number of orders, number of new clients, CPI, month, and year.

As another example of the possibilities of data analysis with this new database, ATAK will be able to understand the types of problems that are causing machine failures by the fail type, average time between failures, and frequency of failure type for each machine token. This would also allow ATAK to utilize the time between failures and correspondingly improve operational efficiency and maintenance scheduling as expressed in the example from Query 2.

These two examples are just the beginning of how a centralized mySQL database will allow ATAK Cosmetics to not only track processes, but also optimize all of the steps of the manufacturing and distribution process. By making data more accessible, it will be easier for the company to apply modeling and prediction techniques, such as the ones previously discussed, to address company needs. Moving towards this kind of data-driven approach allows ATAK Cosmetics to be ahead of their competitors in dealing with the global supply chain crisis that has been exacerbated by COVID-19 throughout the last few years.

There are unlimited future possibilities for the expansion of ATAK Cosmetics with the addition of a refined database. Increased efficiency and organization in terms of tracking, producing, and shipping products will lead to an increase in revenue over time. This growth will not only allow ATAK to expand their team, produce different products and build new factories, but ultimately become the leader in cosmetics manufacturing and exporting in international markets.