University of Toronto
Faculty of Applied Science and Engineering
MIE262 Operations Research I
Warp Shoe - Lab Project

Team Member	**UtorID**	**Student Number**
Dina Aynalem	aynalemd	1006830146
Jonah Ernest	ernestjo	1007065275
Victoria Piroian	piroianv	1006882090

1.0 Abstract

The team was tasked with finding the most profitable production plan for the WARP Shoe Company. The plan is to maximize profit by deciding each x_i, the number of pairs of shoe type i to produce in February 2006. The team first modified the original Microsoft Access database to fit the requirements of the problem. The necessary sets, parameters, objective function, and constraints were then modelled in AMPL to produce code that outputs the optimal profit and production plan. The mathematical model produced an optimal value of $10,998,247.21 and as well as the number of shoes of each type to achieve this profit.

2.0 Introduction

The WARP Shoe Company is an established shoe company in Canada. At the start of 2006, a major competitor of WARP went bankrupt. Consequently, WARP market analysts predicted that the demand for all shoe types would double in February 2006. The management of WARP has tasked the team to find the most profitable production plan for the month. While constructing the production plan, the team must consider several factors dealing with the budget for raw materials, the sales price on shoes (revenue), the cost of not meeting the demand for any type of shoe, the machine working hours restrictions, labour costs, and warehouse capacity limits for holding products. The model does not account for transportation costs and manufacturing sequences. This information was used to formulate a mathematical model and produce an optimal profit and production plan.

3.0 Methodology and Math Model

The following sections will describe the tables used in data importing, the sets, parameters, decision variables, objective function, and constraints defined for this model.

3.1 Database Description

The table below includes the descriptions of all the imported tables from the original Microsoft Access database and indications of which attributes were included and excluded. The descriptions of these tables, their included attributes, and excluded attributes can be found below.

Table 1: Description of WARP Shoe Project Database Tables

Table with Attributes	Description
Product_Master *Attributes:* Product_Num, Product_Name, Sales_Price	The Product_Master table includes the basic product information of each type of shoe pair the WARP Shoe Company manufactures & sells. The Product_Master relation contains the product code for each pair of shoes, the associated product name, and the price for each shoe in ($/pair of shoes). The Product_Master table was imported with the Product_Num and Sales_Price attributes and excluded the Product_Name column. The product

Included Attributes: Product_Num, Sales_Price	number and sales price were included as they provide the exact amount of profit associated with each shoe product, which is needed to establish the decision variables and calculate revenue. The product name was excluded because both the name and number for each product is unique and so only one is necessary.
RM_Master *Attributes:* RM_Num, RM_Name, Cost, S_Quanitity, Category *Included Attributes:* RM_Num, Cost, S_Quantity	The RM_Master table includes the basic raw material information of each type of raw material the WARP Shoe Company uses. The RM_Master relation contains the raw material number for each particular type of raw material, the name of the particular type of raw material, the cost ($/unit) of a particular raw material, the available quantity of each raw material, and the raw material category. The RM_Master table was imported with the RM_Num, Cost, and S_Quanitity attributes and excluded the RM_Name and Category columns. The raw material number and cost were included as they provide the exact cost incurred from each raw material used in the manufacturing process. Further, the available quantity is included because the order up to quantity is needed for a constraint that restricts the amount of raw materials used to be less than the available amount of that specific raw material. The raw material name was excluded because both the name and number are unique for each raw material, and therefore only one is necessary. The category was excluded because it provided no additional information that the team deemed valuable.
Machine_Master *Attributes:* Machine_Num, Machine_Name, Setup_Cost_per_min, OpCost_per_min, Setup_Time_minutes, Section *Included Attributes:* Machine_Num, OpCost_per_min	The Machine_Master table includes the basic machine information of each particular machine. The Machine_Master relation contains the machine number, the machine name, the setup costs ($/minute) for each machine, the operating costs ($/minute) for each particular machine, the amount of minutes to set up a particular machine, and the section where the machine is located. The Machine_Master table was imported with the Machine_Num and OpCost_per_min because it provides the operating cost ($/minute) for each unique machine associated with a distinct machine number. The machine name was excluded because both the name and number are unique for each machine which means one can be neglected because it does not convey any new information. The setup times and costs are negligible as stated in the problem statement, and therefore these two columns are not relevant and were not included in the team's database set.

Warehouse_Master *Attributes:* Warehouse_Num, Location, Capacity, Op_Cost, Distance_to_Plant *Included Attributes:* Warehouse_Num, Op_Cost, Capacity	The Warehouse_Master table includes the basic information about the warehouses. The Warehouse_Master relation contains the unique warehouse number, the location, the holding capacity of the warehouse (in pairs of shoes), the monthly operating cost for the warehouse, and the distance (in kilometres) from the warehouse to the plant. The Warehouse_Master table was imported with the Warehouse_Num, Op_Cost and Capacity. The warehouse number is a unique value that has an associated monthly operating cost and capacity both of which are taken into account in the warehouse capacity constraint. The team excluded the location and distance to the plant columns because the problem statement instructs that transportation costs can be ignored, therefore, distances and locations of the warehouse provide no useful information.
BOM *Attributes:* Product_Num, RM_Num, Quantity *Included Attributes:* Product_Num, RM_Num, Quantity	The BOM table includes the bill of materials - raw materials required per each product. The BOM table contains the product number for each pair of shoes, raw material number, and quantity of raw material required to produce a pair of shoes. The BOM table was imported with Product_Num, RM_Num, and Quantity. The product number and raw material number convey which raw materials are needed to manufacture each shoe type, and the quantity tells how much of each raw material is necessary to produce a single pair of shoes.
Machine_Assign *Attributes:* Product_Num, Machine_Num, Avg_Duration, Step_Num *Included Attributes:* Product_Num, Machine_Num, Avg_Duration	The Machine_Assign table includes the processing time of each shoe on each machine. The Machine_Assign table contains the product number of each shoe type, the machine number, the average duration (in seconds) to process each step number in the processing sequence of each product. The Machine_Assign table was imported with Product_Num, Machine_Num, and Avg_Duration. The step number was excluded because the problem statement states that manufacturing sequence can be ignored. The three other columns express which product (shoe type) is manufactured at which machine and the average duration specifies the average duration to process each step for a specific product on a particular machine.

Demand *Attributes:* Product_Num, Product_Demand *Included Attributes:* Product_Num, Product_Demand	The Demand table includes the product demand of each shoe. The Demand table includes the product number, and the exact demand or amount sold. The Demand table was imported with the product number and product demand. The product number and associated demands were imported because they provide information on the average demands of each store. For the process of predicting the demand across all shoe types for February 2006, see section 5.1.

3.2 Modelling the Sets

The table below includes all sets in the mathematical model along with their descriptions, domains, and some examples.

Table 2: List of Sets Used in the Model

Set	**Description**
set Product_Num;	$1 \leq \ shoe\ type \leq \ 557,\ shoe\ type \in Z_{+}$ The shoe number, i, is the identifier for the shoe a given shoe type. All values if i as described above represent all of the possible types of shoes that can be produced, and are included in the decision variables. For instance, the shoe product numbers have i = {SH001, SH002, …, SH557}.
set Machine_Num;	$1 \leq \ j \leq \ 72,\ j \in Z_{+}$ The set of machine type j, includes all positive integers from 1 to 72 as there are 72 different types of machines used to produce the shoes. Since each shoe consists of multiple parts and there are different machines involved in constructing different types of shoes i, the unique machines must be given a unique identifying number j, where j = {1, 2, …, 72}.
set RM_Num;	$1 \leq \ p \leq \ 165,\ p \in Z_{+}$ The set of raw material number p, includes all positive integers from 1 to 165 as there are 165 different raw materials involved in the production of the 557 shoe types, such as Sole0001, which has a raw material number of p = 1. The raw materials have numbers p = {1, 2, …, 165}.
set Warehouse_Num;	$1 \leq \ k \leq \ 8,\ k \in Z_{+}$

	The set of warehouse numbers k, includes all positive integers from 1 to 8 as there are 8 different warehouses to store shoes. For instance, the Humberline warehouse has k = 1. The warehouses have numbers k = {1, 2, …, 8}.

3.3 Decision Variable(s)

For this integer program, the decision variables are x_i , the number of pairs of shoe type i to produce in February of 2006.

3.4 Modelling the Parameters

The table below includes the descriptions of the parameters that allowed for the simplification of the mathematical model. These parameters are used in both the objective function and the constraints of the mathematical model.

Table 3: Description of Parameters

Parameter	Affiliated Set(s)	Description	Database Table
r_i Sales_Price	Product_Num	The parameter r_i represents the sales price (in $/pair of shoes) for shoe type i. The parameter r_i is used in only the first term of the objective function which represents the revenue produced from shoe sales. This parameter is not used in any constraints but is crucial to the overall revenue term of the objective function as it signifies how much money is made from each sale of a product.	Product_Master
m_j OpCost_per_min	Machine_Num	The parameter m_j is the operating cost (in $/minute) for machine type j. The parameter m_j is used in the second term of the objective function which represents the cost of operating the machines during the month. The parameter is converted to $/second and then multiplied by the time it takes to produce a unit of shoe type i at machine j with the amount of units being	Machine_Master

		manufactured in the month.	
t_{ij} Avg_Duration	Product_Num, Machine_Num	The parameter t_{ij} represents the time (in seconds) required to produce shoe type i by machine j. The parameter is incorporated in the second and fourth terms of the objective function. In the second term, the time it takes to produce shoe type i by machine j is multiplied by operating cost for machine j and the amount of shoes being produced in the month. The fourth term represents the total labour costs of operating the machines, and t_{ij} is multiplied by the number of pairs of shoe type i that are being produced. The parameter t_{ij} is also used in the machine work time constraint.	Machine_Assign
w_k Op_Cost	Warehouse_Num	The parameter w_k is the monthly operating cost for warehouse k. The parameter w_k is used in the third term of the objective function - the monthly warehouse cost. The monthly operating costs w_k , are summed across all warehouses and decrease the overall profit of the WARP Shoe Company. The parameter w_k is not used in any constraints.	Warehouse_Master
n_{ip} Quantity	RM_Num, Product_Num	The parameter n_{ip} represents the quantity of raw material p needed to make one unit of shoe type i. The parameter n_{ip} is used in the last term of the objective function which represents the cost of raw materials to produce all shoes during the month. The parameter n_{ip} is multiplied by the cost per raw material and the number of shoes of shoe type i. The parameter n_{ip} is used in	BOM

		two of the six constraints, specifically the raw material budget and raw material limit constraints. The raw material costs must be less than the raw material budget of $10 million. Further, the quantity of raw material needed to make one unit of shoe type i is restricted to the order up to quantity q_p.	
c_p Cost	RM_Num	The parameter c_p is the cost per unit of raw material p. The parameter c_p is used in the final term of the objective function which represents the cost of raw materials to produce all shoes during the month. The parameter c_p is multiplied by the number of units required to make one pair of shoe type i, with the number of pairs of shoe type i being created each month. The parameter c_p is used in the raw material budget constraint because the cost per raw material must be considered in order to stay within the budget of $10M.	RM_Master
a_k Capacity	Warehouse_Num	The parameter a_k is the monthly holding capacity (in pairs of shoes) of warehouse k. The parameter a_k Is used in the warehouse capacity constraint that restricts the total amount of pair of shoes of type i to be less than the sum of all the warehouses holding capacities. The parameter a_k is not used in the objective function.	Warehouse_Master
q_p S_Quantity	RM_Num	The parameter q_p represents the available quantity of raw material p. The parameter q_p is not used in the objective function but instead in the raw material available quantity limit. The	RM_Master

		raw material quantity limit constraint ensures that the sum of the amount of raw material p multiplied by the number of pairs of shoe type i must be less than the available quantity of that particular raw material.	
d_i Product_Demand	Product_Num	The parameter d_i is the sum across all stores of the average demands for each shoe type for every month between 1997-2003. The parameter d_i is used in the fifth objective function term which addresses the ten-dollar fee incurred when demand is not met. The sum is taken across all the shoe types to ensure that the cost applies for any type of shoe that does not meet its demand. The parameter d_i is also used in the demand limiti constraint which restricts the pairs of shoe type i produced to the sum of the demands of all stores for that specific shoe type. Supply is always less than the demand because it is assumed that the production of anything above the demand is not sold. The reason the parameter d_i is multiplied by two in the constraint is because the WARP market analysts predict the demand to double in the month of February 2006. As the demand doubles from the values in previous years, in all functions and constraints that use this parameter the team will always multiply the demand by 2. In addition, product demand will always be rounded whenever used as it does not make sense to have a fractional amount of demand for a pair of shoes.	Product_Demand

3.5 Modelling the Objective Function

The team decided that the WARP shoe problem would be best represented with an objective function that maximises the profit for the WARP Shoe Company. This can be broken down into maximising the difference between the revenue collected from shoe sales minus all of their associated production costs (see Table 4). The objective function that was created to model this problem is shown below along with a breakdown of each term within it.

$$max\, z = \sum_{i=1}^{557} r_i x_i - \sum_{i=1}^{557}\sum_{j=1}^{72}(\frac{1}{60}m_j)t_{ij}x_i - \sum_{k=1}^{8} w_k - \sum_{i=1}^{557}\sum_{j=1}^{72}\frac{25}{3600}t_{ij}x_i - 10\left(\sum_{i=1}^{557} 2d_i - x_i\right)$$
$$- \sum_{i=1}^{557}\sum_{p=1}^{165} c_p n_{ip} x_i$$

Table 4: Breakdown of Objective Function

Objective Function Term	**Description**
$\sum_{i=1}^{557} r_i x_i$	The first term of the objective function represents the revenue collected from shoe sales (the only source of revenue). This revenue is calculated by multiplying the sales price per pair of shoe type i, by the number of pairs of shoe type i that is going to be produced & sold. If the sum of the revenue collected from selling x_i pairs of shoe type i at a price of r_i across all i, the total revenue for that month is obtained.
$-\sum_{i=1}^{557}\sum_{j=1}^{72}(\frac{1}{60}m_j)t_{ij}x_i$	The second term of the objective functions represents the cost of operating the machines during the month. Since it is a cost, it was subtracted from the objective function. It is composed of the multiplication of the operating cost for machine j (converted to $/second), with the time it takes to produce a unit of shoe type i at machine j, with the number of units of type i that is being produced during the month. This product of three terms is then summed across all of the shoe types being produced across all of the machines that produce that shoe, resulting in the total cost of operating all machines necessary to produce the optimal number of shoes for February 2006.
$-\sum_{k=1}^{8} w_k$	The third term of the objective function represents the monthly warehouse cost. This cost is subtracted from the objective function as it reduces WARP Shoe Company's profit. Since there are eight warehouses, the monthly cost to operate all warehouses is the sum of each warehouse's monthly cost.

$$-\sum_{i=1}^{557}\sum_{j=1}^{72}\frac{25}{3600}t_{ij}x_i$$	The fourth term in the objective function represents the total salary of all employees operating the machines. This cost reduces the profit for the company and is composed of the hourly rate to pay employees, which is \$25/hour. To be consistent with the other units, this was converted into dollars per second by dividing by 3600 seconds/hour. This rate was then multiplied by the time it takes for machine j to produce a pair of shoe type i (which is the same time it takes an employee to make because each machine is operated by one worker), and also by the number of pairs of shoe type i that are being produced. This term is then summed across all machines as each machine is operated by one worker, and then summed across all shoe types to produce the total worker's salaries for the month.
$$-\ 10\left(\sum_{i=1}^{557} 2d_i - x_i\right)$$	The fifth term of the objective function takes into account the ten-dollar fee for not meeting the demand for any type of shoe due to the loss of potential customers. Since it is assumed that the number of pairs of shoe type i is never greater than the demand for that shoe type during any given month, the difference between these terms will always either be zero (if demand is exactly met), or positive if the number of shoe type i is less than the demand for that shoe during the month. Additionally, it is given that the demand for February of 2006 is doubled from previous months, so the demand from the database is doubled. When this term is positive for any shoe type i, the company incurs a cost of ten dollars for each pair, and so it is subtracted from the objective function. The sum is then taken across all of the shoe types to ensure that this cost applies for any type of shoe that does not meet its demand.
$$-\sum_{i=1}^{557}\sum_{p=1}^{165} c_p n_{ip} x_i$$	The final term in the objective function represents the cost of raw materials to produce all shoes during the month, and so it is subtracted from the objective function to reduce profit. It is the product of the cost per unit for raw material p, with the number of that unit required to create one pair of shoe type i, with the number of pairs of shoe type i being produced during that month. The sum is then taken across all types of materials p required to produce all shoe types i. Note that not all raw material types are required for each type of shoe, which is why

	these combinations of i and p are defaulted to 0 in the mathematical program.

3.6 Modelling the Constraints

The table below includes all constraints in the mathematical model, their associated name, and detailed interpretations.

Table 5: List of Constraints With Interpretation

Constraint Name	Constraint	Interpretation
RM_Budget	$\sum_{p=1}^{165}\sum_{i=1}^{557} c_p n_{ip} x_i \leq 10\ 000\ 000$	This constraint originates from the total budget for the raw materials. Since it is given that the company cannot spend more than $10M on raw materials, the product of the cost per raw material unit p with the number of raw material units to produce shoe i with the number of pairs of shoes i to produce must be less than 10 million when summed across all raw materials and shoe types.
Machine_Work_Time	$\sum_{i=1}^{557} t_{ij} x_i \leq 1\ 209\ 600,$ $\forall j = 1,\ldots, 72$	This constraint relates to the maximum operating time for all machines. It is given that each machine can work up to a maximum of 12 hours/day, 28 days/month. This means that the maximum number of seconds that any machine j can operate is 1 209 600 seconds per month. Therefore, this is the upper bound for the product of time (in seconds) to produce a pair of shoes of type i at machine j with the number of pairs of shoes of type i during that month, which is then summed across all shoe types. This constraint applies for all machines (j) since they all have the

		same maximum monthly operating time.
RM_Limit	$\sum_{i=1}^{557} n_{ip} x_i \le q_p$ $\forall\, p = 1,\ldots, 165$	Each raw material, p, has an available quantity, q_p, this is the upper bound for the number of that raw material that can be used to create all the desired shoes. Therefore, the sum of the number of raw material units p to create a pair of shoes of type i multiplied by the number of pairs of shoe type i being produced for all shoe types must be less than the available quantity of that raw material. This constraint applies for all raw material types.
Demand_Limit	$x_i \le 2d_i$, $\forall\, i = 1,\ldots, 557$	This constraint relates to limiting the number of pairs of shoe type i produced for the month. This number has an upper bound of the sum of the demands from all stores for that shoe type. This constraint applies for all shoe types. Essentially, it is not optimal to produce more than demand for that shoe type, as it is assumed that producing more than the demand means that anything above it will not be sold.
Warehouse_Capacity	$\sum_{i=1}^{557} x_i \le \sum_{k=1}^{8} a_k$	Since each warehouse has a holding capacity for the number of pairs of shoes it can store, the sum of holding capacities across all warehouses must be greater than the total number of pairs of shoes produced throughout the month. So the sum of all pairs of shoes of type i summed across all shoe types must be less than the total holding capacity across all of the

		warehouses in order to have enough room to store all the shoes.
Non_Neg	$x_i \geq 0, Z_+$	Since we cannot produce a negative or fraction of a pair of shoes within the month, all of the decision variables, number of pairs of shoe type i to produce, must be nonnegative integers.

3.7 Assumptions and Limitations

The following lists the assumptions and limitations within the model and justifies why they were implemented.

- Since the number of pairs of each shoe type being produced is less than the total demand for that shoe type (because of the demand limit constraint), it is assumed that all shoes being produced in February of 2006 will also be sold during that same month and therefore generating production costs during that time period will also be sold within the same month. Making this assumption allows for the revenue generated from shoe sales to be reflected in the same month that those costs were incurred, and allows the model to neglect the possibility of the difference between the demand for shoe type i and number of pairs of shoe type i being produced being negative (as can be seen in the fifth objective function term to model the fee for not meeting product demand).
- It was assumed that the demand for each shoe type i in February of 2006 is shown in the Demand table of the database. In the original demand table, there was a given demand for each shoe type that was specific to a month, year, and store. In order to predict the demand for February of 2006, the team summed across all stores for the averages across all months for each shoe type and then multiplied by two since it is given that the demand doubles for the month (see section 5.1 for the full method).
- As was stated in the outline, setup times and costs are negligible, transportation costs can be ignored, and the sales prices for each shoe type remains the same for the month.
- A limitation of the model relates to the Machine_Work_Time constraint as it was modelled to only consider the limit on the total number of operating hours in the month, not each individual day. For example, if a machine operated for 20 hours within the month, but all 20 hours were consecutive, the constraint would not be violated despite the machine working past its daily limit.

4.0 Procedure and Results

The following section will describe the structure of the code the team developed to model the LP in AMPL, the issues found with running an IP on AMPL, and the results obtained when the IP was converted to an LP.

4.1 Procedure

The team first modified the provided access database by creating a new table where the demands for each shoe were listed. An explanation of this segment can be found in section 5.1 of the report. The team then created the WS_2022.dat file in AMPL where the team imported all the necessary information from certain tables from the Microsoft Access database. Refer to section 3.1 for a full description of what tables and attributes were imported. The team then created the WS_2022_ORG.mod file where all the sets, decision variables, parameters, objective function, and constraints were defined. Refer to sections 3.2-3.6 respectively for a full overview of these sections. In addition, to answer questions 6 and 7 the team created two additional mod files, WS_2022_Q6.mod, WS_2022_Q7.mod. This was done to account for the differing constraints the three models possessed and to be able to run all three LPs. In the WS_2022_Q6.mod file the only change was a different upper bound for the machine work hours constraint (see section 5.6). The WS_2022_Q7.mod file was created to account for a different upper bound in the raw material budget constraint see section 5.7).

After generating the mathematical model for this problem the team created the WS_2022.run file where the team ran the solution with the files WS_2022_ORG.mod, WS_2022.dat to find the optimal solution and the values, decision variables and profit at this solution. This output was printed to the WS_2022_ORG.out file. The team also answered questions two through five in this code and printed all this to the WS_2022_ORG.out file (Refer to sections 5.2 through 5.5 for the solutions to these questions). In addition, questions 6 and 7 required the team to re-run the model with modified constraints. In order to accomplish this the team reset the model twice, once for each question and ran the solution with the files WS_2022_Q6.mod, WS_2022.dat for question 6, and the files WS_2022_Q7.mod, WS_2022.dat for question 7. The results were outputted to the files WS_2022_Q6.out, and WS_2022_Q7.out, for questions 6 and 7, respectively.

4.2 Conversion from IP to LP

The team originally expelled this mathematical model as an integer program in AMPL, as it is assumed that the company cannot produce a fraction of a pair of shoes and so the decision variables are all integer. This was done by adding the integer definition next to the decision variable x_i. When running the model, it was found that it took over 10 minutes to run, and so in order to be able to acquire an optimal solution within a reasonable amount of time the team made the decision to relax the IP to an LP. This allowed the decision variables to have decimal values, and would result in a much easier model to compute a solution for. For this reason, the program was able to compute an optimal solution to the LP in under less than a minute. It is important to recognize that this change, although it will result in an optimal solution, will not make sense in the real world as there can not be a fractional amount of shoes produced. Section 5.3 describes in more detail how the optimal solution obtained would violate constraints if the values of the decision variables were integers.

4.3 Solution

The solution to the LP modelled in this report had a projected profit of $10,998,247.21 for the optimal solution where 330 of the decision variables had values of zero. The values of all the decision variables and profit at the optimal solution point can be found in the WS_2022_ORG.out file. These characteristics were also evaluated for the new models in questions 6 and 7, and the values of all the decision variables and profit can be found in the output files WS_2022_Q6.out, and WS_2022_Q7.out respectively.

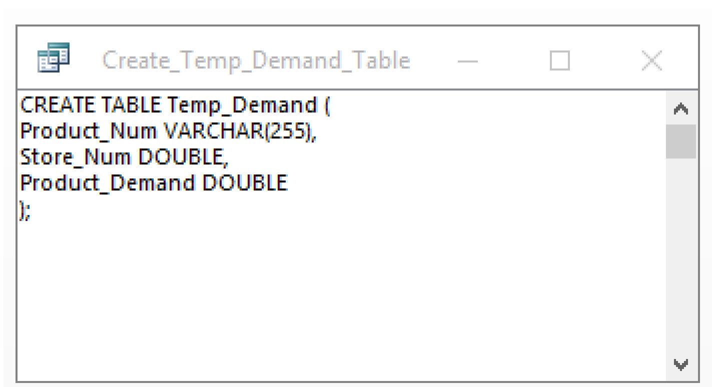
5.0 Questions

As requested in the outline, the following questions have been answered.

5.1 How should you estimate the demand for the month of February?

The demand for the month of February of 2006 for each shoe type i was estimated by finding the demand of each shoe type at each store by taking the average across all the months for the years 1997-2003. Once the average demand for each product at every store was found, the sum across all stores was taken to determine the average total demand for each shoe type i. This process is explained below.

The team used Microsoft Access to create and insert queries that found the average demand for each shoe type at each store. The queries also allowed the team to sum over the demands of all the stores for a particular shoe to obtain the total demand for a given shoe type i. This was done in a series of steps. First, the query ‘Create_Temp_Table_Demand’ created a temporary table, Temp_Demand, with the attributes Product-Num, Store_Num, and Product_Demand. These attributes were of type VARCHAR(255), DOUBLE, and DOUBLE. This query can be seen in Figure 1 below.




```
CREATE TABLE Temp_Demand (
Product_Num VARCHAR(255),
Store_Num DOUBLE,
Product_Demand DOUBLE
);
```

Figure 1: Query for creation of temp-Demand table

The team then created a query to find the average of all the demands for each shoe in every store. This was accomplished using an INNER JOIN on the Product_Num between the two tables Product_Master and Product_Demand and a second INNER JOIN on the Store_Num between the two Store_Master and Product_Demand. This will group the data by shoe ID and by store number. The Avg() function was used to compute the averages of the demands within these groupings. The data in the criteria Product-Num, Store_Num, and Product_Demand was then inserted into the Temp-Demand table. This query is shown in Figure 2.

Add_Demand_Avg_To_Temp_Demand_Table

```
INSERT INTO Temp_Demand ( Product_Num, Store_Num, Product_Demand )
SELECT Product_Master.Product_Num, Store_Master.Store_Num, Avg(Product_Demand.Demand) AS AvgOfDemand
FROM Store_Master INNER JOIN (Product_Master INNER JOIN Product_Demand ON Product_Master.Product_Num =
Product_Demand.Product_Num) ON Store_Master.Store_Num = Product_Demand.Store_Num
GROUP BY Product_Master.Product_Num, Store_Master.Store_Num;
```

Figure 2: Query for inserting data into Temp_Data table from Product_Demand table

The next step was to create the table of demands that will be imported by the AMPL .dat file. The team created a query that creates the table Demand, and has the attributes Product_Num, and Product_Demand with the data types VARCHAR(255) and DOUBLE respectively as shown in Figure 3.

Create_Demand_Table

```
CREATE TABLE Demand (
Product_Num VARCHAR(255),
Product_Demand DOUBLE
);
```

Figure 3: Query for creating Demand table

A fourth query was then used to populate this table with the sum of the values for the Temp_Demand table. Using an INNER JOIN on the Product_Num attribute between the Product_Master and Product_Demand tables. Using the Sum() function the team will obtain the sum of the demands for all stores for a particular shoe i from the Temp_Demand table. This data will then be inserted into the Demand table (Figure 4).

Add_Demands_To_Demand_Table

```sql
INSERT INTO Demand ( Product_Num, Product_Demand )
SELECT Product_Master.Product_Num, Sum(Temp_Demand.Product_Demand) AS
SumOfProduct_Demand
FROM Product_Master INNER JOIN Temp_Demand ON Product_Master.Product_Num =
Temp_Demand.Product_Num
GROUP BY Product_Master.Product_Num;
```

Figure 4: Query for inserting data into Demand table from Temp_Demand table

This process resulted in the Demand with the attributes Product_Num and Product_Demand that will be used in various constraints as well as the objective function (see Figure 5 for the final table). Refer to the results section of the document for the rounding process of the demands in this table.

Demand

Product_Num	Product_Demand
SH001	305.178571428571
SH002	286.583333333333
SH003	288.440476190476
SH004	297.857142857143
SH005	299.988095238095
SH006	297.107142857143
SH007	302.440476190476
SH008	277.761904761905

Figure 5: Final table, Demand, for the demands of each product

5.2 How many variables and constraints do you have?

Since the decision variable x_i is the number of pairs of shoe type i to produce in February of 2006, and there are 557 types of shoes, there are 557 decision variables in the model. After solving the relaxed IP as an LP, it was found that 330 of the decision variables had optimal values of 0, meaning that 330 of the shoe types are not being produced during that month (for the full solution and list of all decision variables refer to the results section). There are six types of constraints included in the model: raw material budget, machine work time, monthly holding capacity for the warehouses, raw material availability, demand limits, and non-negativity. In total, this comes to 1353 constraints (see Figure 6).

```
Question 2:

The number of variables is: _nvars = 557

The number of constraints is: _ncons = 1353
```

Figure 6: Total number of decision variables and constraints in the model

The results shown above are produced using the code in Figure 7.

```
#############################################################################################
printf '\nQuestion 2:\n' > WS_2022_ORG.out;
# question 2 will display the number of variables and constraints in the LP

# print the number of variables in this LP
printf '\nThe number of decision variables is: ' > WS_2022_ORG.out;
# count the numbet of variables in this LP
display _nvars > WS_2022_ORG.out;
# print the number of constraints in this LP
printf 'The number of constraints is: ' > WS_2022_ORG.out;
# count the numbet of constraints in this LP
display _ncons > WS_2022_ORG.out;

#############################################################################################
```

Figure 7: The code for question 2 in the WS_2022.run file

5.3 If you had to relax your integer program to an LP, how many constraints were violated after rounding the LP solution to the closest integer solution?

When attempting to run the integer program for this model, the team found that it would take over 10 minutes to find the optimal solution to this problem. In order to obtain a solution for this model, the team relaxed the IP to a linear program, allowing the decision variables to take on non-negative, non-integer values. Checks were performed to see if the model's six constraints would be violated after rounding the LP solution so that the decision variables were integer, with the rounded values of the optimal solution decision variables (see Figure 8 for the WS_2022.run file for question 3).

```
####################################################################################
printf 'Question 3:\n' > WS_2022_ORG.out;
# question 3 will display the constraints that would have been violated if the LP was an integer program

    # if the cost for the total amount of raw material used for all the shoes
    # is more than thebudget for raw materials,
    # then the raw materials budget constraint is violated
    if sum{i in Product_Num, p in RM_Num} Cost[p]*Quantity[i,p]*round(x[i]) > 10000000 then
        printf '\nThe RM_Budget constraint is violated.' > WS_2022_ORG.out;

    # loop through the machine operating time constraint that exists for each machine j
    for {j in Machine_Num}{
    # if the total operating time for one machine
    # is greater that the amount of time that machine is allowed to operate for,
    # then the machine working time constraint is violated for machine j
        if sum{i in Product_Num} Avg_Duration[i,j]*round(x[i]) > 1209600 then
            printf '\nThe Machine_Work_Time[%s] constraint is violated.', j > WS_2022_ORG.out;
        }

    # loop through the limit of each raw material constraint
    for {p in RM_Num}{
        # if the total amount a raw material used in the production of all shoes i that require this material
        # exceeds the quantity of the raw material available,
        # then the limit of that raw material constraint is violated
        if sum{i in Product_Num} Quantity[i,p]*round(x[i]) > S_Quantity[p] then
            printf '\nThe RM_Limit[%s] constraint is violated.', p > WS_2022_ORG.out;
        }
    # loop through the number of pairs of shoe type i the company sells
    for {i in Product_Num}{
        # if the production of a shoe type i exceeds the demand for that product
        # then the demand constraint is violated
        if round(x[i]) > 2*round(Product_Demand[i]) then
            printf '\nThe Demand_Limit[%s] constraint is violated.', i > WS_2022_ORG.out;
        # if the amount of product i produced is a negative number
        # then the non negativity constraint is violated
        if round(x[i]) < 0 then
            printf '\nThe Non_Neg[%s] constraint is violated.', i > WS_2022_ORG.out;
        }

    # if the total amount of pairs of shoes the company produces is larger than the total warehouse capacity,
    # then the warehouse capacity cpnstraint is violated
    if sum{i in Product_Num} round(x[i]) > sum {k in Warehouse_Num} Capacity[k] then
        printf '\nThe Warehouse_Capacity constraint is violated.' > WS_2022_ORG.out;

####################################################################################
```

Figure 8: The code for question 3 in the WS_2022.run file

It was found that the only type of constraint violated was the RM_Limit constraint. There was a total of 86 constraints that were violated by rounding the LP decision variables to integer values. For the full list of the constraints that were violated, refer to the WS_2022_ORG.out file.

5.4 Which constraints are binding, and what is the real-world interpretation of those binding constraints?

Three out of the six types of constraints are binding constraints. There were 98 raw material limit (RM_Limit), 62 demand limit (Demand_Limit), and 330 non-negativity binding constraints. Overall, 490 out of the total 1353 constraints are binding. These binding constraints were identified by determining if the slack of a given constraint was equal to zero, as this is the definition of a binding constraint (see Figure 9 for the WS_2022.run file for question 4). For the full list of the binding constraints see the WS_2022_ORG.out file.

```
####################################################################################
printf '\n\nQuestion 4:\n' > WS_2022_ORG.out;
# question 4 will print all the binding constraints in this LP

printf '\nThe binding constraints for this optimal solution are:\n' > WS_2022_ORG.out;

    # if the slack of the RM_Budget constraint is 0
    # then this constraint is binding
    if RM_Budget.slack == 0 then
        printf '\t RM_Budget constraint is binding.\n' > WS_2022_ORG.out;

    # loop through the machine operating time constraint that exists for each machine j
    for {j in Machine_Num}{
        # if the slack of the Machine_Work_Time constraint is 0
        # then this constraint is binding
        if Machine_Work_Time[j].slack == 0 then
            printf '\t Machine_Work_Time[%s] constraint is binding.\n', j > WS_2022_ORG.out;
        }

    # loop through the max quantity of raw materials constraint that exists for each raw material p
    for {p in RM_Num}{
        # if the slack of the RM_Limit constraint is 0
        # then this constraint is binding
        if RM_Limit[p].slack == 0 then
            printf '\t RM_Limit[%s] constraint is binding.\n', p > WS_2022_ORG.out;
        }
    # loop through the number of shoe type i the company sells
    for {i in Product_Num}{
        # if the slack of the Demand_Limit constraint is 0
        # then this constraint is binding
        if Demand_Limit[i].slack == 0 then
            printf '\t Demand_Limit[%s] constraint is binding.\n', i > WS_2022_ORG.out;
        # if the slack of the Non_Neg constraint is 0
        # then this constraint is binding
        if Non_Neg[i].slack == 0 then
            printf '\t Non_Neg[%s] constraint is binding.\n', i > WS_2022_ORG.out;
        }
    # if the slack of the Warehouse_Capacity constraint is 0
    # then this constraint is binding
    if Warehouse_Capacity.slack == 0 then
        printf '\t Warehouse_Capacity constraint is binding.\n' > WS_2022_ORG.out;

####################################################################################
```

Figure 9: The code for question 4 in the WS_2022.run file

The RM_Limit constraints can be interpreted in the way that for All of the 98 RM_Limit binding constraints can be interpreted as there are 98 different raw material types (p) that are being fully depleted. In other words, the optimal solution is completely using 98 different raw materials, and the shoe types requiring any of the 98 raw materials involved in their associated binding constraints cannot be produced any more as there are no more available resources.

All of the 62 demand binding constraint limits can be interpreted as 62 of the different shoe types being produced at their maximum production limit. If the solution were to suggest producing more than the current optimal solution, the additional produced units would go unsold as it is surpassing the demand for that given product. Therefore these additional units would be incurring additional costs without generating any revenue.

The 330 non-neg binding constraint can be interpreted in the way that there are 330 out of the total 557 shoe types that are not being produced. Each of the decision variables that take on a value of zero lead to a non-negativity binding constraint because they are the lower bound of the non-neg constraint. Since we cannot produce a negative number of shoes, the decision variables being bound by this constraint take on a value of zero and are not being produced in the optimal solution.

5.5 Assume that some additional warehouse space is available at the price of $10/box of shoes. Is it economical to buy it? What is the optimal amount of space to buy in this situation?

Since the Warehouse_Capacity constraint was found to be non-binding, there still exists additional warehouse space that is not occupied in the current optimal solution (see Figures 10 and 11 for the code and output of the non-binding warehouse capacity constraint).

```
###########################################################################################
printf '\nQuestion 5:\n' > WS_2022_ORG.out;
# question 5 will determine if the warehouse capacity constraint is binding or not
# it will help answer the question of whether or not it is optimal to increase the warehouse capacity

    # if the slack of the Warehouse_Capacity constraint is not 0
    # then this constraint is not binding
    if Warehouse_Capacity.slack != 0 then
            printf '\t Warehouse_Capacity constraint is not binding.\n\n' > WS_2022_ORG.out;

###########################################################################################
```

Figure 10: The code for question 5 in the WS_2022.run file

Therefore, since the total warehouse space currently has slack, there is no need to incur additional costs for more space as it will not be used. The optimal amount of additional space to buy in this situation is none.

```
Question 5:
        Warehouse_Capacity constraint is not binding.
```

Figure 11: The output for question 5 in the WS_2022_ORG.out file

5.6 Imagine that machines were available for only 8 hours per day. How would your solution change? Which constraints are binding now? Does the new solution seem realistic to you?

The team found that the solution did not change as the constraint for the number of hours per day a machine can be operated was not binding in the original model, where machines operated for 12 hours per day. The team created a second model where the maximum number of hours that any machine j could work over its total production of shoes must remain under 806400 seconds. These seconds accounted for each machine working 8 hours a day for all 29 days of the month of

February. This change did not result in a difference in profit or in a difference of decision variable values. The binding constraints of this LP are different from the previous model. Here there are 480 binding constraints, but they are the same types of constraints as in the original model; 88 raw material limit (RM_Limit), 62 demand limit (Demand_Limit), and 330 non-negativity binding constraints. This solution does seem realistic as the profit at the optimal solution of this LP did not change from that of the previous model. This is because the constraint who's upper bound was changed was not originally a binding constraint so increasing it would not produce a change in the objective function or decision variable values. For a full list of the binding constraints refer to the WS_2022_Q6.out file.

```
subject to Machine_Work_Time{j in Machine_Num}: sum{i in Product_Num} Avg_Duration[i,j]*x[i] <= 806400;
# constraint on max monthly machine operating time
```

Figure 12: The code for the machine working hours constraint when modified so machines work 8 hours per day in the WS_2022_Q6.mod file

5.7 If in addition there were a $7,000,000 budget available to buy raw materials, what would you do? Change your formulation and solve again.

It was found that the solution did not change when the raw material budget was increased from $10,000,000 to $17,000,000. The team created a third model where the maximum amount of money that can be spent by the company on the raw materials is $17,000,000. This change did not make a difference in the profit or in a difference of decision variable values. The binding constraints of this LP are different from the previous model. Here there are 487 binding constraints, but they are the same types of constraints as in the original model; 95 raw material limit (RM_Limit), 62 demand limit (Demand_Limit), and 330 non-negativity binding constraints. As the raw material budget constraint was not a binding constraint in the original model, meaning that there was still extra money to spend on raw materials, an extra $7,000,000 will not change the solution of the model. This will simply cause there to be an extra $7,000,000 to spend on raw materials left over after the LP is optimised. In this case there is no gain from this addition to the budget of raw materials, and so the team is indifferent in whether they would take or decline this sum as it will not change the profit at the optimal solution. For a full list of the binding constraints refer to the WS_2022_Q7.out file.

```
subject to RM_Budget: sum{i in Product_Num, p in RM_Num} Cost[p]*Quantity[i,p]*x[i] <= 17000000;
# constraint for the budget of raw materials
```

Figure 13: The code for the raw material budget constraint when modified so raw materials have a total budget of $17,000,000 in the WS_2022_Q7.mod file

6.0 Conclusion

In summary, the team was able to provide a viable solution to the WARP Shoe Company's management team. The optimal profit value produced by the LP model is $10,998,247.21, and 330 of the decision variables in this solution had values of zero. In order to attain this optimal solution value and production plan, the team first modified the original Microsoft Access database by making a new table for the product demand information. All necessary information was imported into AMPL and then all sets, decision variables, parameters, objective function, and constraints were defined. The values of all decision variables and profit can be found in the output files WS_2022_Q6.out, and WS_2022_Q7.out respectively. Through sensitivity analysis, the team was able to determine that the optimal amount of additional space to buy at $10/unit is none for this specific situation. Further, the team identified that the solution would not change if the available machine operating hours were decreased to 8 hours a day because it was not binding in the original model. Last, the addition of $7M to the raw material budget was found to have no effect on the optimal profit amount or in the decision variable values. The mathematical model and results provide the team with full confidence in recommending the production plan outlined previously in the report to the WARP Shoe Company because it produces an optimal profit value of $10,998,247.21.