Online Auction Database Project
IEOR 115 || Professor Ken Goldberg || Fall 2014
Shirley Bao, Jessica Huang, Jie Li, Cindy Mo, Jeremy Wan, Yijian Wu, Kangting Yu, Aurona Zhang
Sellegit Background

- Downtown Berkeley Startup
- Conventional Auction Business Model
- Founded by Berkeley Alumni – Go Bears!!

Online Marketplace for Safe Transactions near College Campuses

Sellegit

Background | Founders | Schema | Relationships | EER | Queries | Normalization
Database Schema

1. Visit(VisitID, User\(^2\), IPaddress, loginTime, duration, numClick)
2. User(UID, frequency, device)
   a. Visitor(UID\(^2\), ...)
   b. RegisteredUser(UID\(^2\), Fname, Lname, email, phoneNumber, sellegitCredits, referralCode, password, university\(^6\))
      i. Buyer(UID\(^2\), ...)
      ii. Seller(UID\(^2\), ...)
3. Employee(EID, Fname, Lname, email, phoneNumber, title, DOB)
4. Item(IID, listPrice, mostRecentBidPrice, status, numBids, description, datePosted, minPrice, condition, vendor\(^8\), location\(^9\), order\(^10\), seller\(^2b(ii)\))
5. ItemType(TID, item\(^4\), typeName, makeYear, material, brand, madeIn, course\(^7\), faceValue, detailedDescription)
6. University(univName, startDate, endDate, size, location\(^9\))
7. Course(CoID, cName, cNum, semester, year, location\(^9\))
8. Vendor(VID, vName, industry, size, location\(^9\))
9. Location(zipCode, city, state, country)
Database Schema

10. Order(OID, time, paymentType, paymentAmt, pickupTime, location, item, buyer, seller)
11. MarketingEvent(MID, name, date, platform, type)
12. Comment(CID, time, content, item, user)
13. Feedback(FID, time, content, rating, buyer, seller)
14. Inventory(InvID, location, stockDate, size, clearBy)
15. Generates(user, marketingEvent)
16. Organizes(employee, marketingEvent)
17. Bids(buyer, item)
18. Receives(seller, order)
19. Browses(user, item)
20. IsOfType(item, type)
21. Stores(inventory, item)
22. Stocks(vendor, inventory)
23. RelatesTo(textbook, course)
24. OfferedAt(university, course)
QUERIES
Query 1: Listing Price Recommendation

At what price should a seller list his or her item to ensure a quick sell?

- Determine face value of item
- Determine item’s category
- Calculate past items’ percentage change

Apply percentage to current item
Query 1: Listing Price Recommendation

Business Justification:

- Helps give users **peace of mind** when listing and pricing an item for auction
- Ensures **increased probability of selling** the item based on previous sales of similar type
- Provides **flexibility to the seller** to choose an appropriate listing price based on our recommendation
Query 1: Listing Price Recommendation

Implementation Process:

- **SQL**
  - Extracts relevant data from the database

- **SAS**
  - Run statistical analysis on past pricing trends

- **MS Access**
  - Displays the appropriate results based on the SQL queries
Query 1: Listing Price Recommendation

SQL Code:

```sql
SELECT Order.OID, Order.original_price, Order.final_price, Order.IID AS Order_IID,
       ItemType.IID AS ItemType_IID, ItemType.TypeName,
       Round((((Order.original_price-Order.final_price)/Order.original_price), 4) AS percentage_change
FROM [Order] INNER JOIN ItemType ON Order.IID = ItemType.IID
ORDER BY ItemType.TypeName DESC, Round((((Order.original_price-Order.final_price)/Order.original_price), 4) DESC;
```
Query 1: Listing Price Recommendation

Sample Output:

<table>
<thead>
<tr>
<th>OID</th>
<th>original_price</th>
<th>final_price</th>
<th>Order_IID</th>
<th>ItemType_IID</th>
<th>TypeName</th>
<th>percentage_c</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>199</td>
<td>90.01</td>
<td>6</td>
<td>6 game</td>
<td></td>
<td>0.5477</td>
</tr>
<tr>
<td>24</td>
<td>99.99</td>
<td>62</td>
<td>24</td>
<td>24 game</td>
<td></td>
<td>0.3799</td>
</tr>
<tr>
<td>32</td>
<td>5</td>
<td>4.94</td>
<td>32</td>
<td>32 game</td>
<td></td>
<td>0.012</td>
</tr>
<tr>
<td>45</td>
<td>10</td>
<td>0.25</td>
<td>45</td>
<td>45 furniture</td>
<td></td>
<td>0.975</td>
</tr>
<tr>
<td>31</td>
<td>15</td>
<td>1.05</td>
<td>31</td>
<td>31 furniture</td>
<td></td>
<td>0.93</td>
</tr>
<tr>
<td>14</td>
<td>28</td>
<td>5.01</td>
<td>14</td>
<td>14 furniture</td>
<td></td>
<td>0.8211</td>
</tr>
<tr>
<td>42</td>
<td>80</td>
<td>15</td>
<td>42</td>
<td>42 furniture</td>
<td></td>
<td>0.8125</td>
</tr>
<tr>
<td>41</td>
<td>80</td>
<td>15</td>
<td>41</td>
<td>41 furniture</td>
<td></td>
<td>0.8125</td>
</tr>
<tr>
<td>48</td>
<td>238</td>
<td>45</td>
<td>48</td>
<td>48 furniture</td>
<td></td>
<td>0.8109</td>
</tr>
<tr>
<td>17</td>
<td>40</td>
<td>10</td>
<td>17</td>
<td>17 furniture</td>
<td></td>
<td>0.75</td>
</tr>
<tr>
<td>39</td>
<td>60</td>
<td>16</td>
<td>39</td>
<td>39 furniture</td>
<td></td>
<td>0.7333</td>
</tr>
<tr>
<td>38</td>
<td>69.98</td>
<td>20.02</td>
<td>38</td>
<td>38 furniture</td>
<td></td>
<td>0.7139</td>
</tr>
<tr>
<td>46</td>
<td>50</td>
<td>20</td>
<td>46</td>
<td>46 furniture</td>
<td></td>
<td>0.6</td>
</tr>
<tr>
<td>10</td>
<td>69</td>
<td>29.93</td>
<td>10</td>
<td>10 furniture</td>
<td></td>
<td>0.5662</td>
</tr>
<tr>
<td>11</td>
<td>69</td>
<td>29.93</td>
<td>11</td>
<td>11 furniture</td>
<td></td>
<td>0.5662</td>
</tr>
<tr>
<td>23</td>
<td>149</td>
<td>70</td>
<td>23</td>
<td>23 furniture</td>
<td></td>
<td>0.5302</td>
</tr>
<tr>
<td>1</td>
<td>45</td>
<td>21.29</td>
<td>1</td>
<td>1 furniture</td>
<td></td>
<td>0.5269</td>
</tr>
</tbody>
</table>
Query 1: Listing Price Recommendation

SAS Code:

```sas
libname exdat excel 'C:\Users\student\Desktop\query1full.xlsx';
data full; set exdat.'Order Query$'n;
proc import out = full
datafile = 'C:\Users\student\Desktop\query1full.xlsx'
dbms = excel replace;
range = 'Order Query$'n;
run;

proc means data = full mean median q1 q3 std;
class typeName;
var percentage_change;
title '';

data furniture; set full;
if typeName = 'furniture';
run;

proc univariate data = furniture noprint;
var percentage_change;
output out=percnt pctlpts = 5 50 95 pctlpre = P;
title "5 50 95 percentile of percentage change for funitures";
proc print data = percnt;
ods graphics off;
symbol v=plus;
title 'Normal Quantile-Quantile Plot for Percentage Change';
proc capability data = furniture noprint;
   spec lsl = 0.3 usl = 0.8;
   qqplot percentage_change;
run;
proc univariate data = furniture;
histogram percentage_change / normal
   midpoints = (0.1 to 0.9 by 0.1);
title "Histogram of Percentage Change(Furnitures)";
run;
```
Query 1: Listing Price Recommendation

SAS Output:

The MEANS Procedure
Analysis Variable: percentage_change percentage_change

<table>
<thead>
<tr>
<th>TypeName</th>
<th>N Obs</th>
<th>Mean</th>
<th>Median</th>
<th>Lower Quartile</th>
<th>Upper Quartile</th>
<th>Std Dev</th>
</tr>
</thead>
<tbody>
<tr>
<td>book</td>
<td>3</td>
<td>0.3737333</td>
<td>0.3743000</td>
<td>0.2500000</td>
<td>0.4969000</td>
<td>0.1234510</td>
</tr>
<tr>
<td>clothing</td>
<td>4</td>
<td>0.4131000</td>
<td>0.4701000</td>
<td>0.1915000</td>
<td>0.6347000</td>
<td>0.3019002</td>
</tr>
<tr>
<td>electronic</td>
<td>11</td>
<td>0.4975000</td>
<td>0.5000000</td>
<td>0.2744000</td>
<td>0.6000000</td>
<td>0.2400388</td>
</tr>
<tr>
<td>furniture</td>
<td>29</td>
<td>0.5351759</td>
<td>0.5268000</td>
<td>0.3681000</td>
<td>0.7333000</td>
<td>0.2412260</td>
</tr>
<tr>
<td>game</td>
<td>3</td>
<td>0.3132000</td>
<td>0.3799000</td>
<td>0.0120000</td>
<td>0.5477000</td>
<td>0.2740078</td>
</tr>
</tbody>
</table>
Query 1: Listing Price Recommendation

SAS Output:

The UNIVARIATE Procedure
Variable: percentage_change  (percentage_change)

Moments

<table>
<thead>
<tr>
<th>N</th>
<th>Sum Weights</th>
</tr>
</thead>
<tbody>
<tr>
<td>29</td>
<td>15.5201</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mean</th>
<th>0.53517586</th>
</tr>
</thead>
<tbody>
<tr>
<td>Std Deviation</td>
<td>0.24122562</td>
</tr>
<tr>
<td>Skewness</td>
<td>-0.21103571</td>
</tr>
<tr>
<td>Uncorrected SS</td>
<td>9.93530267</td>
</tr>
<tr>
<td>Coeff Variation</td>
<td>0.0741569</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Variance</th>
<th>0.05818999</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kurtosis</td>
<td>-0.2483729</td>
</tr>
<tr>
<td>Corrected SS</td>
<td>1.62931977</td>
</tr>
<tr>
<td>Std Error Mean</td>
<td>0.04479455</td>
</tr>
</tbody>
</table>

Basic Statistical Measures

<table>
<thead>
<tr>
<th>Location</th>
<th>Variability</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.53517586</td>
</tr>
<tr>
<td>Median</td>
<td>0.526800</td>
</tr>
<tr>
<td>Mode</td>
<td>0.500000</td>
</tr>
<tr>
<td>Std Deviation</td>
<td>0.24123</td>
</tr>
<tr>
<td>Variance</td>
<td>0.05819</td>
</tr>
<tr>
<td>Range</td>
<td>0.94550</td>
</tr>
<tr>
<td>Interquartile Range</td>
<td>0.36520</td>
</tr>
</tbody>
</table>

Tests for Location: Mu0=0

<table>
<thead>
<tr>
<th>Test</th>
<th>-Statistic-</th>
<th>-----p Value-----</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student’s t</td>
<td>11.94734</td>
<td>Pr &gt;</td>
</tr>
<tr>
<td>Sign</td>
<td>14.5</td>
<td>Pr &gt;=</td>
</tr>
<tr>
<td>Signed Rank</td>
<td>217.5</td>
<td>Pr &gt;=</td>
</tr>
</tbody>
</table>

Quantiles (Definition 5)

<table>
<thead>
<tr>
<th>Quantile</th>
<th>Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>100% Max</td>
<td>0.8750</td>
</tr>
<tr>
<td>99%</td>
<td>0.8750</td>
</tr>
<tr>
<td>95%</td>
<td>0.8300</td>
</tr>
<tr>
<td>90%</td>
<td>0.8211</td>
</tr>
<tr>
<td>75% Q3</td>
<td>0.7333</td>
</tr>
<tr>
<td>50% Median</td>
<td>0.5268</td>
</tr>
<tr>
<td>25% Q1</td>
<td>0.3681</td>
</tr>
<tr>
<td>10%</td>
<td>0.1767</td>
</tr>
<tr>
<td>5%</td>
<td>0.0507</td>
</tr>
<tr>
<td>1%</td>
<td>0.0295</td>
</tr>
<tr>
<td>0% Min</td>
<td>0.0295</td>
</tr>
</tbody>
</table>

Background | Founders | Schema | Relationships | EER | Queries | Normalization
Query 1: Listing Price Recommendation

SAS Output:

![Histogram of Percentage Change(Furnitures)](image-url)
Query 2: Item Recommendation

What other listed items would a Sellegit user be interested in purchasing?

What items did other users browse?

Count items in common

Recommend items to user
Query 2: Item Recommendation

Business Justification:

- Extends the time a user stays on the website by providing many **relevant and eye-catching listings**
- **Increases the turnover rate** of listed items which provides incentive for sellers to use site
- Promotes **user loyalty** by automatically displaying other wanted items before leaving the website
Query 2: Item Recommendation

Implementation Process:

**SQL**
Extracts browsing and item data from the database

**MS Access**
Displays the appropriate results based on the SQL queries
Query 2: Item Recommendation

SQL Code:

```
SELECT item.IID, item.Name AS SuggestedItem
FROM item,
     (SELECT top 5 B.IID, COUNT(B.IID) AS C FROM Browses B, (SELECT B.UID FROM Browses B WHERE B.IID = 1) AS P WHERE B.UID = P.UID GROUP BY B.IID ORDER BY count(B.IID) DESC) AS itemIID
WHERE ((item.IID)=[itemIID].[IID]);
```

Sample Output:

```
<table>
<thead>
<tr>
<th>IID</th>
<th>SuggestedItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IKEA single studying desk, black, wood</td>
</tr>
<tr>
<td>13</td>
<td>IKEA MALM 6-drawer Chest</td>
</tr>
<tr>
<td>12</td>
<td>21 forever girls shoes</td>
</tr>
<tr>
<td>34</td>
<td>IKEA Sultan Holmsta Full Size Spring Mattress, deliverable</td>
</tr>
<tr>
<td>37</td>
<td>Walmart single sofa, clean</td>
</tr>
</tbody>
</table>
```
Query 2: Item Recommendation

Sample Output:

<table>
<thead>
<tr>
<th>IID</th>
<th>SuggestedItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>IKEA single bed frame, deliverable</td>
</tr>
<tr>
<td>45</td>
<td>Storage</td>
</tr>
<tr>
<td>37</td>
<td>Walmart single sofa, clear</td>
</tr>
<tr>
<td>12</td>
<td>21 forever girls shoes</td>
</tr>
<tr>
<td>33</td>
<td>Free Queen Size Mattress Giveaway</td>
</tr>
<tr>
<td>22</td>
<td>IKEA full size mattress, extra firm, 22' thick</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>IID</th>
<th>SuggestedItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>Walmart plastic chair</td>
</tr>
<tr>
<td>4</td>
<td>IKEA wood chair,</td>
</tr>
<tr>
<td>45</td>
<td>Storage</td>
</tr>
<tr>
<td>23</td>
<td>Walmart full size mattress, extra firm, 18' thick, clearn</td>
</tr>
<tr>
<td>50</td>
<td>Canon Rebel T3i DSLR Camera</td>
</tr>
</tbody>
</table>
Query 2: Item Recommendation

**Aztec Print Cross Body Bag**

Pick up near

Original Price: $22.00

$5.00

Make Offer (0)

**Requirement**

Jessica Kim

**DESCRIPTION**

Item Specifics
Bought at Target

Recommendations

People are also viewing:

- [Used Item](#)
- [Used Item](#)
- [Used Item](#)
Query 3: Marketing Effort Optimization

When is the best time during the semester to deploy a Marketing Campaign for Sellegit?

- Fit a time series model to site traffic data
- Predict future site activity using model
- Select appropriate phases for potential Marketing Campaigns
Query 3: Marketing Effort Optimization

Business Justification:

✓ Utilize web traffic logs to model user activity

✓ Analyze the effectiveness of past marketing campaigns using the established model

✓ Aid in strategically timing their marketing efforts to minimize wasteful campaigns
Query 3: Marketing Effort Optimization

Implementation Process:

**SQL**
Extracts web traffic data from the database

**MS Access**
Displays the web traffic results based on the SQL queries

**R**
Fit a time series model to web traffic data
Query 3: Marketing Effort Optimization

SQL Code:

```
SELECT Visit.LoginDate, Count(Visit.VisitID) AS Number_of_visitor_per_day
FROM Visit
GROUP BY LoginDate;
```

Sample Output:

```
<table>
<thead>
<tr>
<th>LoginDate</th>
<th>Number_of_visitor_per_day</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fri Apr 25 2014</td>
<td>443</td>
</tr>
<tr>
<td>Sat Apr 26 2014</td>
<td>350</td>
</tr>
<tr>
<td>Thu Apr 24 2014</td>
<td>207</td>
</tr>
</tbody>
</table>
```
### Query 3: Marketing Effort Optimization

**R Code:**

```r
# Load data
sellegit = read.csv("~/Sellegit_timeseries.csv", header = FALSE)
y = sellegit[,2]

# Plot the original data
plot(y, type = 'l', col = 'red', xlab = 'month', ylab = 'number_of_visitors_per_day', xaxt = 'n')
axis(1, at = c(7,40,64,124,154,184,214,244,264,294,324), label = c("Nov", "Dec", "Jan", "Feb", "March", "April", "May", "Jun", "Jul", "Aug", "Sep", "Oct"))
title("Sellegit Number of Visitors Over Months (Raw Data)"

# Remove the fluctuation by taking 5 Day Moving Average
f10 = rep(1/5, 5)
y_lag = filter(y, f10, sides = 1)
plot(y_lag, col = "blue", xlab = 'month', ylab = 'number_of_visitors_per_day', xaxt='n')
axis(1, at = c(7,40,64,124,154,184,214,244,264,294,324), label = c("Nov", "Dec", "Jan", "Feb", "March", "April", "May", "Jun", "Jul", "Aug", "Sep", "Oct"))
title("Sellegit Number of Visitors Over Months (5 Day Moving Average)"

# Fit 2 half-semester data into seasonal AR(1) model
Spring_first_half = y_lag[74:145]
Fall_first_half = y_lag[294:365]
model_series_1 = c(Spring_first_half, Fall_first_half)
tsdDisplay(model_series_1)
mod1 = sarima(model_series_1, 0,1,0,1,0,72)
```

**Queries**

3. Fit 2 half-semester data into seasonal AR(1) model
   - Remove the fluctuation by taking 5 Day Moving Average
   - Plot the original data

**Background | Founders | Schema | Relationships | EER | Queries | Normalization**
Query 3: Marketing Effort Optimization

R Output:

Seasonal AR(1) time series model:

\[ \hat{Y}_t = \varphi_0 + \varphi_1 \cdot Y_{t-a} + e \]

Coefficients:

<table>
<thead>
<tr>
<th></th>
<th>Coefficients</th>
<th>s.e.</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>sar1</td>
<td>0.6312</td>
<td>0.0664</td>
<td>90.6249</td>
</tr>
<tr>
<td>constant</td>
<td>3084.9986</td>
<td>90.6249</td>
<td></td>
</tr>
</tbody>
</table>

Time series model according to our analysis:

\[ \hat{Y}_t = 0.6312 + 3085 \cdot Y_{t-a} \]
Query 3: Marketing Effort Optimization

R Output:

Sellegit Number of Visitors Over Months (Raw Data)

![Graph showing the number of visitors per day over different months.](image-url)
Query 3: Marketing Effort Optimization

R Output:

**Sellegit Number of Visitors Over Months (5_day Moving Average)**

- X-axis: month
- Y-axis: number_of_visitors_per_day
- Data points from Nov to Oct
- Graph shows trends in visitor numbers over time.
Query 3: Marketing Effort Optimization

Prediction Output:

Sellegit Web Visitor Prediction for Spring 2015

Days of Semester

Number of Visitors
Query 4: User Retention Assessment

How likely will any given user continue to use Sellegit’s service after a certain time period?

- Calculate retention rate from 2013 to 2014
- Determine survival probability for 2015
- Analyze how many users bought 1, 2, 3... items last year
Query 4: User Retention Assessment

Business Justification:

- Forecast future user activity to help **plan company strategy** for expansion and publicity
- **Reduce client loss** by targeting users who are at the greatest risk of leaving
- **Identify key events** caused the most increases and decreases in user activity
Query 4: User Retention Assessment

Implementation Process:

SQL
Calculate retention rate from 2013 to 2014

MS Access
Result from SQL query is the input for the Kaplan-Meier Estimators model

Python
Calculate survival probability based on model
Query 4: User Retention Assessment

SQL Code:

```
SELECT Count([Order].UID) AS Expr1
FROM [Order]
WHERE ((([Order].Year)=13));

SELECT Count([Order].UID) AS Expr1
FROM [Order]
WHERE ((([Order].Year)=13));

SELECT Start.Expr1 AS Start, Stay.Expr1 AS Stay,
([Stay]/[Start])*100 AS RetentionPercentage
FROM Start, Stay;

SELECT Order.UID, Count(Order.UID) AS CountOfUID
FROM [Order]
WHERE (((Order.Year)=14))
GROUP BY Order.UID;

SELECT HowManyOrderPerUID.CountOfUID AS NumItemsSoldPerPerson, Count(*) AS NumPeople
FROM HowManyOrderPerUID
GROUP BY HowManyOrderPerUID.CountOfUID;
```

Sample Output:

```
<table>
<thead>
<tr>
<th>Start</th>
<th>Stay</th>
<th>RetentionPercentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>53</td>
<td>41</td>
<td>77.3584905660377</td>
</tr>
</tbody>
</table>
```

```
<table>
<thead>
<tr>
<th>NumItemsSoldPerPerson</th>
<th>NumPeople</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>58</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td>6</td>
<td>1</td>
</tr>
</tbody>
</table>
```
Query 4: User Retention Assessment

Python Code:

```python
In [1]:
import numpy as np
import pandas as pd
import xld

data=pd.ExcelFile('Desktop/Shared/sf_Documents/purchase.xlsx')
df=data.parse('Sheet1',

def compute(dataFrame):
    size=list(dataFrame['purchase size'])
    loss=list(dataFrame['number of people'])
    initial=sum(loss)
    SiLoRiSe=zip(size,loss)
    SiLoRiSe=sorted(SiLoRiSe)
    SiLoRiSe[0]=SiLoRiSe[0](initial,(initial-SiLoRiSe[0][1])/float(initial),)
    i=1
    while i<len(SiLoRiSe):
        risk_set=SiLoRiSe[i-1][2]-SiLoRiSe[i-1][1]
        loss_set=SiLoRiSe[i][1]
        servival=SiLoRiSe[i][1]*(risk_set-loss_set)/float(risk_set)
        SiLoRiSe[i]=SiLoRiSe[i]+(risk_set,servival,)
        i+=1
        servival_dict={
            for element in SiLoRiSe:
                servival_dict['S('+str(element[0])+')']={element[3]
        return servival_dict

out=compute(df)
out.DataFrame(out, index=[1])
```
### Query 4: User Retention Assessment

Python Output:

```
Out[1]:

<table>
<thead>
<tr>
<th></th>
<th>S(1)</th>
<th>S(2)</th>
<th>S(3)</th>
<th>S(4)</th>
<th>S(5)</th>
<th>S(6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.246753</td>
<td>0.168831</td>
<td>0.12987</td>
<td>0.090909</td>
<td>0.012987</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Probability of a user purchasing 1 item in 2015
Probability of a user purchasing 2 items in 2015
Probability of a user purchasing 3 items in 2015
Probability of a user purchasing 4 items in 2015
Probability of a user purchasing 5 or 6 items in 2015
Query 5: Optimal Warehouse Location

If Sellegit were to build warehouses to store products, where should they build them?

Pinpoint Sellegit user hotspots

Minimize distance between hotspots

Plot these optimal locations
Query 5: Optimal Warehouse Location

Implementation Process:

SQL
Extracts order data from the database

R
Calculate distance and test matrices; create XML input file

AMPL
Find optimal solution

Google Earth
Plot order and warehouse locations
Business Justification:

- Optimize company operations by finding the **best central locations** to build warehouses.

- Gain information for **scaling up the company** in future expansion efforts.

- Provide **valuable geographical information** about users that can further help in marketing.
Query 5: Optimal Warehouse Location

SQL Code:

```sql
SELECT city, lati, long
FROM order;
```

Sample Output:

<table>
<thead>
<tr>
<th>city</th>
<th>lati</th>
<th>long</th>
</tr>
</thead>
<tbody>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.89182</td>
<td>-122.27581</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.86938</td>
<td>-122.27298</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.87005</td>
<td>-122.27848</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.89902</td>
<td>-122.25918</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.88541</td>
<td>-122.25373</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.89512</td>
<td>-122.26083</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.89595</td>
<td>-122.27167</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.8665</td>
<td>-122.27807</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.89064</td>
<td>-122.267</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.90056</td>
<td>-122.26156</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.88279</td>
<td>-122.2767</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.86845</td>
<td>-122.25764</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.88709</td>
<td>-122.26303</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.87925</td>
<td>-122.25835</td>
</tr>
<tr>
<td>Berkeley, CA, USA</td>
<td>37.8809</td>
<td>-122.2634</td>
</tr>
</tbody>
</table>
Query 5: Optimal Warehouse Location

R Code:

Load in data to current environment

```
order.data <- read.csv("C:/Users/jie/Desktop/order data.csv")
x = order.data$long
y = order.data$lati
unique(order.data$city)
```

Retrieve the relevant subset of data

```
point = matrix(NA, 183, 2)
point[,1] = x
point[,2] = y
```

Deleting missing data

```
data <- na.omit(point)
```

Set up initial warehouse locations

```
# set up initial points and plot warehouse location
k = matrix(c(-122.24, 37.78, -122.27, 37.83, -122.28, 37.88, -122.42, 
37.80, -122.46, 37.78, -122.34, 37.92, -122.2, 37.8, -122.03, 37.53, 
-121.96, 37.52), 9,2, T)
```
Query 5: Optimal Warehouse Location

R Code:

```r
earthdist = function(long1, lati1, long2, lati2) {
  dlong = long2 - long1
  dlati = lati2 - lati1
  a = (sin(pi*dlati/360))^2 + cos(pi*lati1/180) * cos(pi*lati2/180) * (sin(pi*dlong/360) ^2)
  c = 2 * atan2(sqrt(a), sqrt(1-a))
  earthdist = 3961 * c
  return(earthdist)
}

test = matrix(0, 9, 183) #binary 0/1
rownames(test) = c("l1","l2","l3", "l4", "l5", "l6", "l7", "l8", "l9")
dist = matrix(NA, 9, 183) #distance
rownames(dist) = c("l1","l2","l3", "l4", "l5", "l6", "l7", "l8", "l9")

for(i in 1:183) {
  dis = earthdist(k[1:9,1], k[1:9,2], data[i,1], data[i,2])
  dist[,i] = t(matrix(dis))
  test[which(dis == min(dis)),i] = 1
}
```
Query 5: Optimal Warehouse Location

R Output:

<table>
<thead>
<tr>
<th>I</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>9</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>I</th>
<th>V1</th>
<th>V2</th>
<th>V3</th>
<th>V4</th>
<th>V5</th>
<th>V6</th>
<th>V7</th>
<th>V8</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>4.28551924</td>
<td>2.72729333</td>
<td>2.80718657</td>
<td>4.80792808</td>
<td>3.93222424</td>
<td>4.52946289</td>
<td>4.56019881</td>
<td>2.5615032</td>
</tr>
<tr>
<td>3</td>
<td>0.84852341</td>
<td>0.82811989</td>
<td>0.69285125</td>
<td>1.73760707</td>
<td>1.4813907</td>
<td>1.47870838</td>
<td>1.19265445</td>
<td>0.93921269</td>
</tr>
<tr>
<td>6</td>
<td>4.00685568</td>
<td>5.06111634</td>
<td>4.81545619</td>
<td>4.64074393</td>
<td>5.2786789</td>
<td>4.64832417</td>
<td>4.08112198</td>
<td>5.00949064</td>
</tr>
<tr>
<td>9</td>
<td>30.9693233</td>
<td>29.6060439</td>
<td>29.8185758</td>
<td>30.8919912</td>
<td>29.9378461</td>
<td>30.7121172</td>
<td>31.081689</td>
<td>29.6068069</td>
</tr>
</tbody>
</table>

Test Matrix

Distance Matrix
Query 5: Optimal Warehouse Location

Optimizing Process in AMPL:

- Model File
- Data File

Solution!
Query 5: Optimal Warehouse Location

Model File Code:

\[
\begin{align*}
\text{Min} & \quad \sum_{i=1}^{9} \sum_{j=1}^{183} x_i \cdot dist[i, j] \cdot test[i, j] \\
\text{S.T} & \quad \sum_{i=0}^{9} x_i = 3
\end{align*}
\]

param location > 0 integer; # number of location
param order > 0 integer; # number of order
param dist{1..location, 1..order} >= 0;
param test{1..location, 1..order} >= 0;

var x{i in 1..location} binary; # = 1 if warehouse i is ok, 0 otherwise

minimize distance: sum{i in 1..location, j in 1..order} x[i] * dist[i,j] * test[i,j];
subject to number: sum{i in 1..location} x[i] = 3;
Query 5: Optimal Warehouse Location

Data File Code Snippet:

```
param location := 9;
param order := 183;

param dist: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31
32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47
48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63
64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79
80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95
96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111
112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127
128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143
144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159
160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175
176 177 178 179 180 181 182 183 :=
10.14493599 2.382578525 2.382578525 2.382578525 2.382578525 2.382578525 2.382578525 2.382578525
2.187317146 18.19185877 8.866741391 12.27323943 1.026889825 1.026889825 1.026889825 1.026889825
2.382578525 8.137862374 11.229129 9.899121978 9.446551974 2.382578525 2.382578525 2.382578525
13.355904451 2.167229789 2.382578525 2.382578525 2.382578525 2.382578525 2.382578525 2.382578525
```
### Query 5: Optimal Warehouse Location

Data File Code Snippet:

```plaintext
<table>
<thead>
<tr>
<th>param test:</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
<th>11</th>
<th>12</th>
<th>13</th>
<th>14</th>
<th>15</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16</td>
<td>17</td>
<td>18</td>
<td>19</td>
<td>20</td>
<td>21</td>
<td>22</td>
<td>23</td>
<td>24</td>
<td>25</td>
<td>26</td>
<td>27</td>
<td>28</td>
<td>29</td>
<td>30</td>
</tr>
<tr>
<td></td>
<td>32</td>
<td>33</td>
<td>34</td>
<td>35</td>
<td>36</td>
<td>37</td>
<td>38</td>
<td>39</td>
<td>40</td>
<td>41</td>
<td>42</td>
<td>43</td>
<td>44</td>
<td>45</td>
<td>46</td>
</tr>
<tr>
<td></td>
<td>48</td>
<td>49</td>
<td>50</td>
<td>51</td>
<td>52</td>
<td>53</td>
<td>54</td>
<td>55</td>
<td>56</td>
<td>57</td>
<td>58</td>
<td>59</td>
<td>60</td>
<td>61</td>
<td>62</td>
</tr>
<tr>
<td></td>
<td>64</td>
<td>65</td>
<td>66</td>
<td>67</td>
<td>68</td>
<td>69</td>
<td>70</td>
<td>71</td>
<td>72</td>
<td>73</td>
<td>74</td>
<td>75</td>
<td>76</td>
<td>77</td>
<td>78</td>
</tr>
<tr>
<td></td>
<td>80</td>
<td>81</td>
<td>82</td>
<td>83</td>
<td>84</td>
<td>85</td>
<td>86</td>
<td>87</td>
<td>88</td>
<td>89</td>
<td>90</td>
<td>91</td>
<td>92</td>
<td>93</td>
<td>94</td>
</tr>
<tr>
<td></td>
<td>96</td>
<td>97</td>
<td>98</td>
<td>99</td>
<td>100</td>
<td>101</td>
<td>102</td>
<td>103</td>
<td>104</td>
<td>105</td>
<td>106</td>
<td>107</td>
<td>108</td>
<td>109</td>
<td>110</td>
</tr>
<tr>
<td></td>
<td>112</td>
<td>113</td>
<td>114</td>
<td>115</td>
<td>116</td>
<td>117</td>
<td>118</td>
<td>119</td>
<td>120</td>
<td>121</td>
<td>122</td>
<td>123</td>
<td>124</td>
<td>125</td>
<td>126</td>
</tr>
<tr>
<td></td>
<td>128</td>
<td>129</td>
<td>130</td>
<td>131</td>
<td>132</td>
<td>133</td>
<td>134</td>
<td>135</td>
<td>136</td>
<td>137</td>
<td>138</td>
<td>139</td>
<td>140</td>
<td>141</td>
<td>142</td>
</tr>
<tr>
<td></td>
<td>144</td>
<td>145</td>
<td>146</td>
<td>147</td>
<td>148</td>
<td>149</td>
<td>150</td>
<td>151</td>
<td>152</td>
<td>153</td>
<td>154</td>
<td>155</td>
<td>156</td>
<td>157</td>
<td>158</td>
</tr>
<tr>
<td></td>
<td>160</td>
<td>161</td>
<td>162</td>
<td>163</td>
<td>164</td>
<td>165</td>
<td>166</td>
<td>167</td>
<td>168</td>
<td>169</td>
<td>170</td>
<td>171</td>
<td>172</td>
<td>173</td>
<td>174</td>
</tr>
<tr>
<td></td>
<td>176</td>
<td>177</td>
<td>178</td>
<td>179</td>
<td>180</td>
<td>181</td>
<td>182</td>
<td>183</td>
<td>183</td>
<td>183</td>
<td>183</td>
<td>183</td>
<td>183</td>
<td>183</td>
<td>183</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

Background | Founders | Schema | Relationships | EER | Queries | Normalization
Query 5: Optimal Warehouse Location

AMPL Solution Output:

```ampl
AMPL: reset;
AMPL: model model.txt;
AMPL: data data.txt;
AMPL: solve;
MINOS 5.51: ignoring integrality of 9 variables
MINOS 5.51: optimal solution found.
5 iterations, objective 40.44984533
AMPL: display x;
x [*] :=
  1 1
  2 0
  3 0
  4 0
  5 1
  6 0
  7 1
  8 0
  9 0
;
```
Query 5: Optimal Warehouse Location

R Code for XML Input:

```r
lDoc <- newXMLDoc() # set up XML documentoot <- newXMLNode("kml", doc = doc, namespaceDefinitions = "http://www.opengis.net/kml/2.2")
name <- newXMLNode("name", "Sellegit", parent = Document)
description <- newXMLNode("description",
  "Optimized Locations", parent = Document)
for (i in 1:length(latitude)) {
  placemark <- newXMLNode("Placemark", parent = Document)
  point <- newXMLNode("Point", parent = placemark)
  coordinates <- newXMLNode("coordinates",
    paste(longitude[i], ",", latitude[i], sep = ""),
    parent = point)
  #timestamp <- newXMLNode("TimeStamp", parent = placemark)
  #when <- newXMLNode("when", time[i],
    #parent = timestamp)
}
```

Build the XML node tree skeleton

Populate each coordinate into the above XML tree

Background | Founders | Schema | Relationships | EER | Queries | Normalization
Query 5: Optimal Warehouse Location

XML File Snippet:

```xml
<?xml version="1.0"?>
<kml xmlns="http://www.opengis.net/kml/2.2">
<Document>
  <name>Sellegit</name>
  <description>Optimized Locations</description>
  <Style id="pushpin">
    <IconStyle id="mystyle">
      <Icon>
        <href>http://maps.google.com/mapfiles/kml/paddle/W.png</href>
        <scale>50</scale>
      </Icon>
    </IconStyle>
  </Style>
  <Style id="optimal">
    <IconStyle id="mystyle">
      <Icon>
        <href>http://maps.google.com/mapfiles/kml/pal2/icon10.png</href>
        <scale>8.0</scale>
      </Icon>
    </IconStyle>
  </Style>
  <Placemark>
    <Point>
      <coordinates>122.27581,37.89182</coordinates>
    </Point>
  </Placemark>
  <Placemark>
    <Point>
      <coordinates>122.27298,37.86938</coordinates>
    </Point>
  </Placemark>
</Document>
</kml>
```
Query 5: Optimal Warehouse Location

XML File Coordinates → Plot in Google Earth

......
Query 5: Optimal Warehouse Location
Query 5: Optimal Warehouse Location
NORMALIZATION
Decomposing to 1NF

Original Relation:

\[ \text{ItemType}(TID, IID, \text{typeName}, \text{makeYear}, \text{material}, \text{brand}, \text{madeIn}, \text{CoID}, \text{faceValue}, \text{detailedDescription}) \]

To normalize to 1NF:

A item may be made from many materials, so material is a multi-valued attribute. Let’s remove it!

- \[ \text{ItemType}(TID, IID, \text{typeName}, \text{makeYear}, \text{brand}, \text{madeIn}, \text{CoID}, \text{faceValue}, \text{detailedDescription}) \]
- \[ \text{Item_Material}(IID, \text{material}) \]
Decomposing to 2NF

1NF from previous slide:
- ItemType(TID, IID, typeName, makeYear, brand, madeIn, CoID, faceValue, detailedDescription)
- Item_Material(IID, material)

To normalize to 2NF:
We can know makeYear, brand, madeIn, CoID, faceValue, and detailedDescription from IID.
- ItemType(TID, IID, typeName)
- Item_Material(IID, material)
- ItemInfo(IID, makeYear, brand, madeIn, CoID, faceValue, detailedDescription)
Decomposing to 3NF & BCNF

2NF from previous slide:
- ItemType(TID, IID, typeName)
- Item_Material(IID, material)
- ItemInfo(IID, makeYear, brand, madeIn, CoID, faceValue, detailedDescription)

To normalize to 3NF:
No transitive dependencies!

To normalize to BCNF:
TID → TypeName
- ItemType(IID, TID)
- Item_Material(IID, material)
- ItemInfo(IID, makeYear, brand, madeIn, CoID, faceValue, detailedDescription)
Decomposing to 3NF & BCNF

Original Relation:
Course(CoID, cName, cNum, semester, year, univName)
- Already in 1NF & 2NF

To normalize to 3NF:
cNum, semester, year, univName → cName
- Course(CoID, cNum, semester, year, univName)

To normalize to BCNF:
cNum, semester, year, univName → CoID
- Course(cNum, semester, year, univName)
THANK YOU!

Jeff Zhang
CEO of Sellegit

Peter Chen
Co-Founder of Sellegit

Ken Goldberg
Our FAVORITE Professor!!

Animesh Garg
Our AWESOME GSI!!

Have a wonderful Winter Break &
Good Luck with Finals!