Exercise 1.1

The number of association rules is calculated by the formula $3^d - 2^{(d+1)} + 1$
Proof:
If we have d items then we can form $2^d - 1$ subsets and each of these sets can be mapped with multiple other subsets and in particular each subset can be mapped with subsets that dont contain any similar items so for a subset with length K the number of subsets that can be mapped to it is $2^{(d-k)} - 1$
Now we have to sum all possibilities for that

$R = \sum_{k=1}^{d} \binom{d}{k}(2^{(d-k)} - 1) = \sum_{k=1}^{d} \binom{d}{k}2^{(d-k)} - \sum_{k=1}^{d} \binom{d}{k}$

Since $\sum_{k=1}^{d-1} \binom{d}{k} = 2^d - 1$ therefore

$R = \sum_{k=1}^{d} \binom{d}{k}2^{(d-k)} - 2^d + 1$

Since $3^d = \sum_{k=1}^{d} \binom{d}{k}2^{d-k} + 2^d$

then $\sum_{k=1}^{d} \binom{d}{k}2^{d-k} = 3^d - 2^d$

So $R = 3^d - 2^d - (2^d) + 1 = 3^d - 2^{(d+1)} - 1$

Exercise 1.2

First we find all frequent items for k =1

item	count
A	1
C	2
D	1
E	4
K	5
M	3
N	2
O	3
U	1
Y	3

So F1 = {E, K, M, O, Y}

Now we find C2 from F1

C2 = {Ek, EM, EO, EY, KM, KO, KY, MO, MY, OY}

And we check each itemset to see if it is frequent

So F2 = {Ek, EO, KM, KO, KY} because all other itemsets have frequency <3

Now we find C3 from F2, and we we can only form EKO from EK,EO,KO

C3 = {EKO}

And we check this itemset to see if it is frequent

and we notice that it is frequent as it appears 3 times
So F3 = {EKO}

We cant form any candidates anymore and so all frequent items are

F = {E, K, M, O, Y, EK, EO, KM, KO, KY, EKO}

To prove the correctness, we need to prove the Apriori algorithem is sound and complete.

It's sound since each itemset in F_i has been checked for freqency.

We can prove the alg. is complete by induction:

1. We have all 1-itemsets are in C_1 and therefore all frequent $1-itemsets$ are in F_1.
2. Assume all frenquent n-itemsets are in F_n for $n \in N$
3. $n \rightarrow n+1$

   let X be a frequent $(n+1)$-itemsets

   all n-itemsets of X are in F_n and frenqent, so $X \in C_{n+1}$, $X \in F_{n+1}$

To prove the Irredundency:

Let X be a frequent n-itemset

Then $X \notin \{F_1, \ldots, F_{n-1}, F_{n+1}, \ldots\}$, but $X \in F_n$,

Assume X appears twice in F_n

So X will appears twice in $C_n \Rightarrow$ some (n-1)-subset appears twice in F_{n-1}

In this way, some 1-itemset will appears twice in F_1 and C_1

It is contradictory with $C_1 = I$ (which is a set each item only appear once) .

According to above provement we can say the Apriori algorithem correctly and irredundantly generates all frequent itemsets.

Exercise 4

Claim

The Apriori algorithm enumerates the set of frequent itemsets in incremental polynomial time.

Proof

In the first iteration $\mathcal{F}_1$ gets printed because $\mathcal{C}_1 = \mathcal{I}$ and searching for all frequent 1-itemsets takes $|D|$ time. Now in general checking whether a k-itemset is frequent or not can be done in $|D|$ time and in each iteration this has to be done for all k-itemsets in $\mathcal{C}_k$ hence it is polynomial in $|D|$ and $|\mathcal{C}_k|$. So in order to show if each print has polynomial delay we have to proof that **CANDIDATE-GENERATION** is polynomial.
The **CANDIDATEGENERATION** algorithm get $\mathcal{F}_k$ of frequent k-itemsets as an input and return candidates $\mathcal{C}_{k+1}$. Lets note $l = \mathcal{F}_k$ the cardinality. We loop $\mathcal{O}(l^2)$ times over over pair X, Y of k-itemsets that differ only in the last element. Creating the set Z by concatinating strings has polynomial time complexity. And then we check if all k-subsets of Z are in $\mathcal{F}_k$ and this can be done in at most polynomial time in l since the cardinality of C_{k+1} is polynomial in the cardinality of $\mathcal{F}_k$
$\Rightarrow$ The Apriori algorithm is incremental polynomial.

Exercise 1.5

k = 2

$F_2 = EK$
$c(E \rightarrow K) = 1 \geq 0.8$, so print $E \rightarrow K$
$c(K \rightarrow E) = 4/5 = 0.8$, so print $K \rightarrow E$
$H_1 = \{K, E\}$
GenerateRules stops early since $k = 2 = 1 + 1$

$F_2 = EO$
$c(E \rightarrow O) = 3/4 < 0.8$
$c(O \rightarrow E) = 1 \geq 0.8$, so print $O \rightarrow E$
$H_1 = \{E\}$
GenerateRules stops early since $k = 2 = 1 + 1$

$F_2 = KM$
$c(K \rightarrow M) = 3/5 < 0.8$
$c(M \rightarrow K) = 1 \geq 0.8$, so print $M \rightarrow K$
$H_1 = \{K\}$
GenerateRules stops early since $k = 2 = 1 + 1$

$F_2 = KO$
$c(K \rightarrow O) = 3/5 < 0.8$
$c(O \rightarrow K) = 1 \geq 0.8$, so print $O \rightarrow K$
$H_1 = \{K\}$
GenerateRules stops early since $k = 2 = 1 + 1$

$F_2 = KY$
$c(K \rightarrow Y) = 3/5 < 0.8$
$c(Y \rightarrow K) = 1 \geq 0.8$, so print $Y \rightarrow K$
$H_1 = \{K\}$
GenerateRules stops early since $k = 2 = 1 + 1$

k = 3

$F_3 = EOK$
$c(EK \rightarrow O) = 3/4 < 0.8$
$c(OK \rightarrow E) = 1 \geq 0.8$, so print $OK \rightarrow E$
$c(EO \rightarrow K) = 1 \geq 0.8$, so print $EO \rightarrow K$
$H_1 = \{E, K\}$
$k = 3 > 2 = m + 1$, so GenerateRules continues
$H_2 = \{EK\}$
$c(O \rightarrow EK) = 1 \geq 0.8$, so print $O \rightarrow EK$
Recursive execution of GenerateRules stops early since $k = 3 = 2 + 1$

Printed rules: $E \rightarrow K, K \rightarrow E, O \rightarrow E, M \rightarrow K, O \rightarrow K, Y \rightarrow K, OK \rightarrow E, EO \rightarrow K, O \rightarrow EK$