

# HowMessy



## How Messy is Your Database

	<u>Page</u>
■ How messy is your database?	2
■ Hashing algorithm	5
■ Interpreting master dataset lines	12
■ Master dataset solutions	15
■ HowMessy sample report (detail dataset)	17
■ Repacking a detail dataset	22
■ Detail dataset solutions	26
■ Estimating response time	29
■ Automating HowMessy analysis	30
■ Summary	33

# *How messy is your database?*

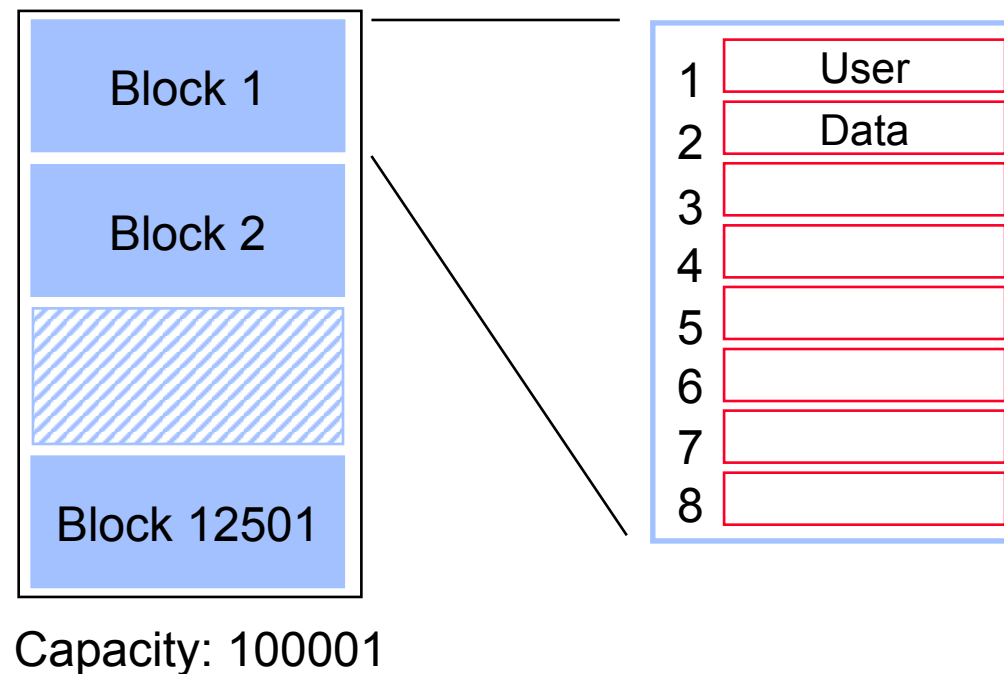


- A database is messy if it takes more I/O than it should
- Unnecessary I/O is still a major limiting factor even on MPE/iX machines
- Databases are messy by nature
- Run HowMessy or DBLOADNG against your database
  - HowMessy is a bonus program for Robelle customers
  - DBLOADNG is a contributed library program

# Blocks



- TurboIMAGE does all I/O operations in blocks
- A block may contain many user records
- More entries per block means fewer I/Os
- Fewer I/Os means better performance



# Record location in masters

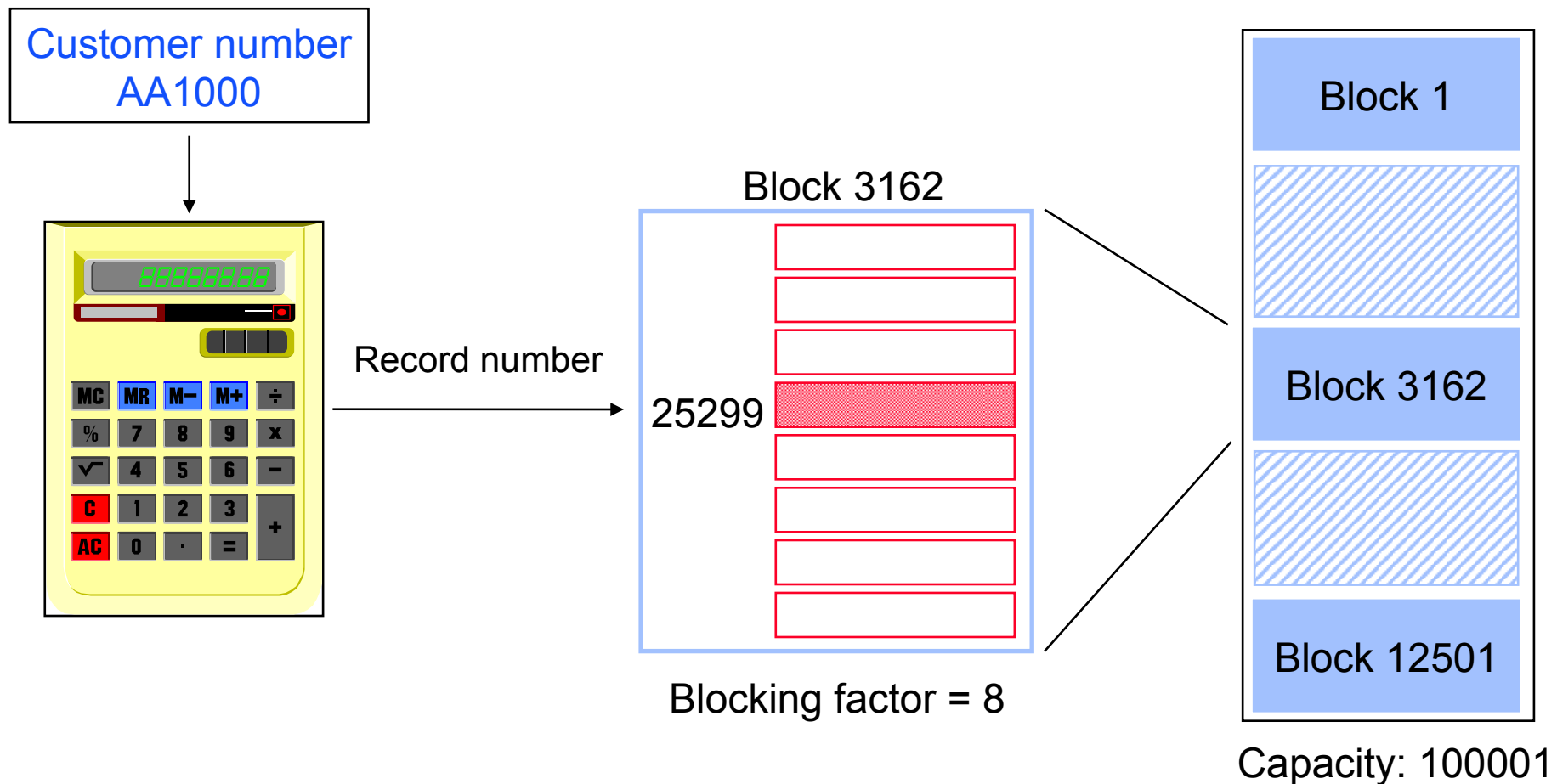


- Search item values must be unique
- Location of entries is determined by a hashing algorithm or a primary address calculation
- Calculation is done on search item value to transform it into a record number between one and the capacity
- Different calculation depending on the search item type
  - X, U, Z, and P give random results
  - I, J, K, R, and E give predictable results



# Hashing algorithm

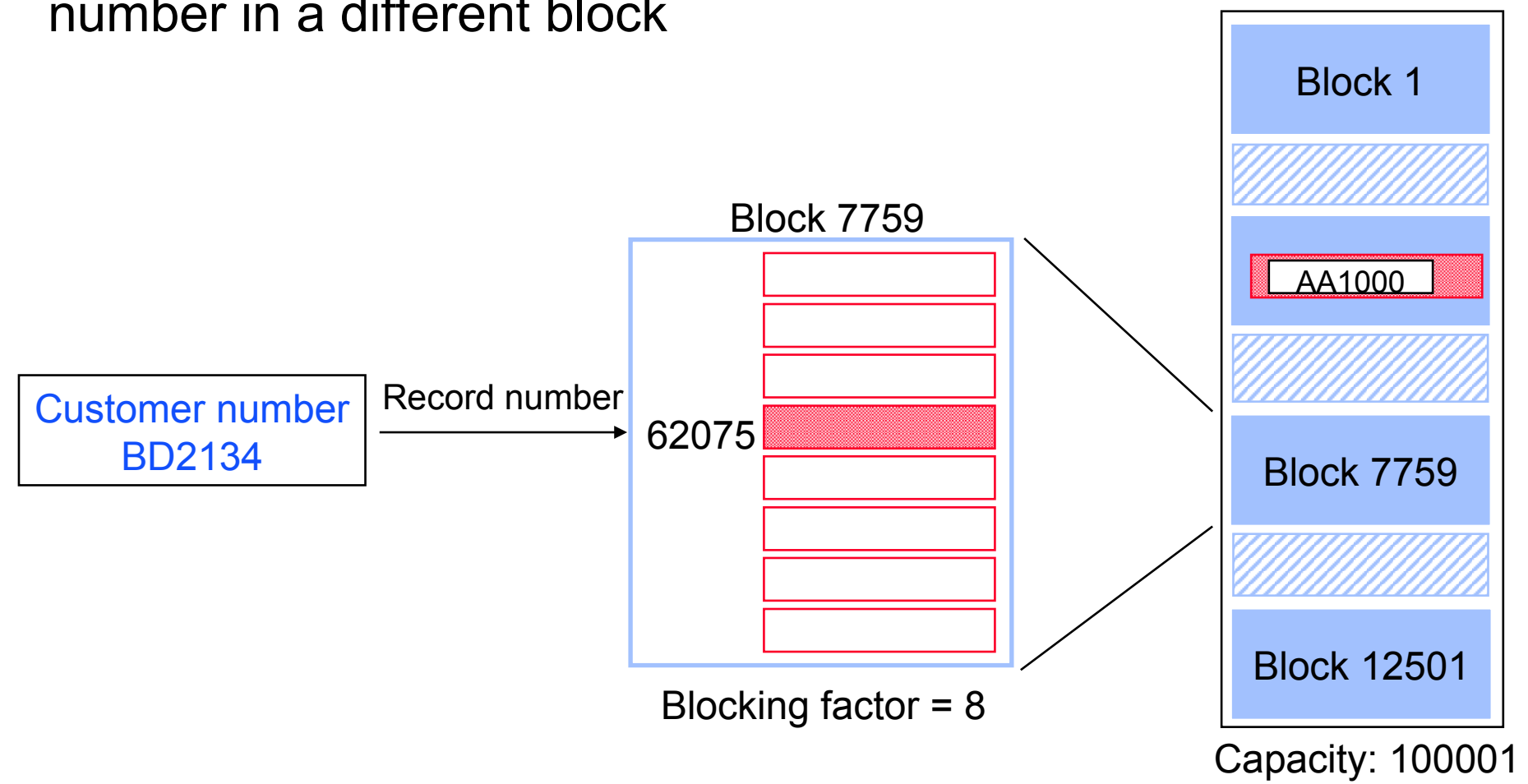
- Customer number AA1000 is transformed into a record number





# Hashing algorithm (no collision)

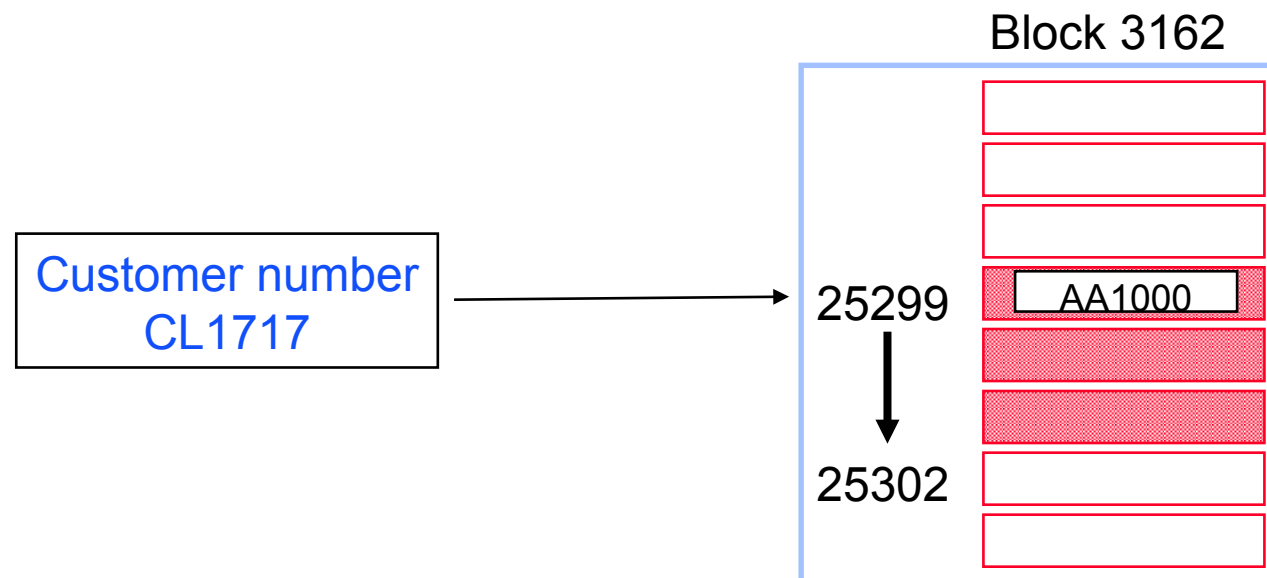
- Customer number BD2134 gives a different record number in a different block



# Hashing algorithm (collision - same block)



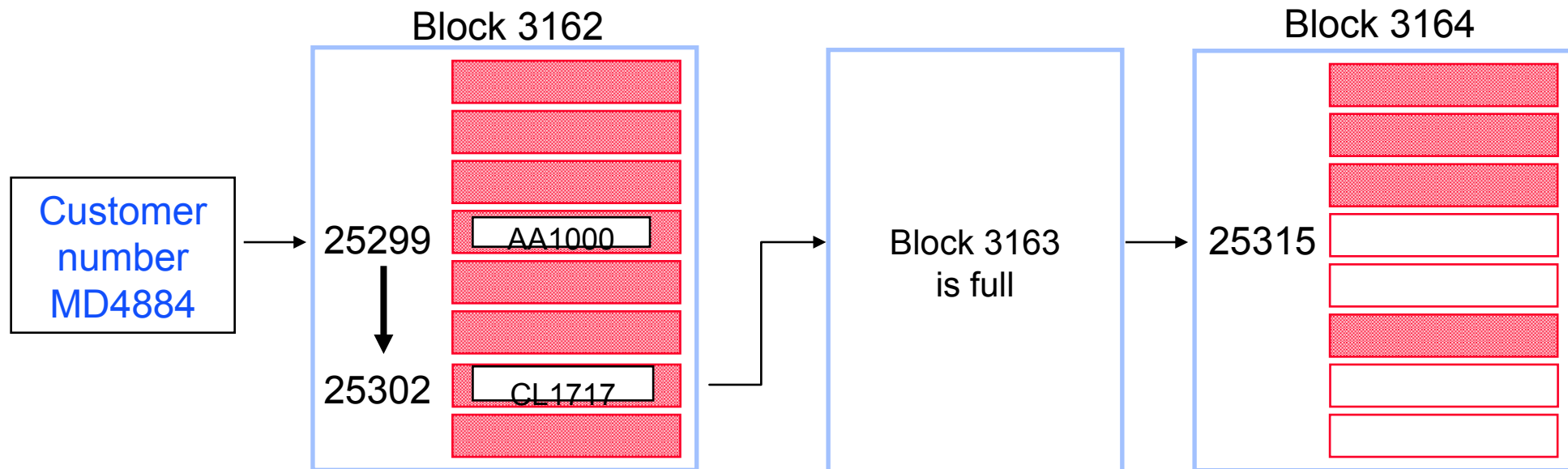
- Customer number CL1717 hashes to the same record number as AA1000 location
- TurboIMAGE tries to find an empty location in the same block. If it finds one, no additional I/O is required.
- CL1717 becomes a secondary entry. Primary and secondary entries are linked using pointers that form a chain.



# Hashing algorithm (collision - different block)

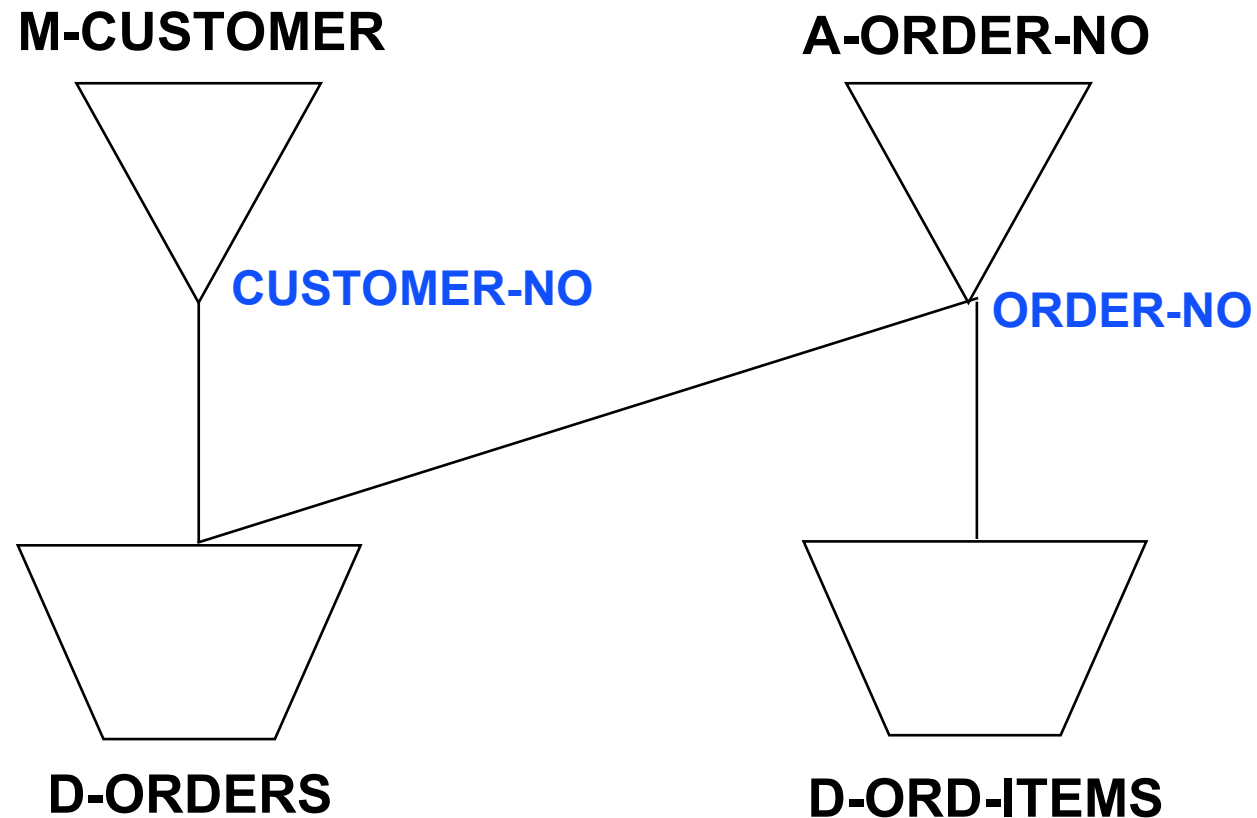


- Customer number MD4884 collides with AA1000
- No more room in this block. TurboIMAGE reads the following blocks until it finds a free record location.
- In this case, MD4884 is placed two blocks away, which requires two additional I/Os.





# An example TurboIMAGE database



# HowMessy sample report



HowMessy/XL (Version 2.2.1)  
TurboIMAGE/3000 databases

Data Base: STORE.DATA.INVENT  
By Robelle Consulting Ltd.

Run on: MON, JAN 9, 1995, 11:48 AM  
Page: 1

Secon- Max Type Data Set	Load daries	Blks	Blk Capacity EntriesFactor (Highwater)	Fact
M-Customer	Man	248113	17801871.7%30.5% 1496	11
A-Order-No	Ato	1266783	76855660.7%25.7% 1	70
D-Orders	Det	1000000	76855876.9%( 851445)	32
D-Ord-Items	Det	4000000	345851186.5%( 3470097)	23

	Max	Ave	Std	Expd	Avg	Ineff	Elong-
Search Field	Chain	Chain	Dev	Blocks	Blocks	Ptrs	ation
Customer-No	32	1.92	0.32	1.00	1.90	90.5%	1.90
Order-No	10	1.35	0.62	1.00	1.00	0.0%	1.00
!Order-No	1	1.00	0	1.00	1.00	0.0%	1.00
S Customer-No	80	14.34	17.76	1.75	9.20	57.2%	5.25
S !Order-No	1604	8.06	35.75	1.36	11.32	72.5%	8.34

# HowMessy sample report (master dataset)



HowMessy/XL (Version 2.2.1)  
TurboIMAGE/3000 databases

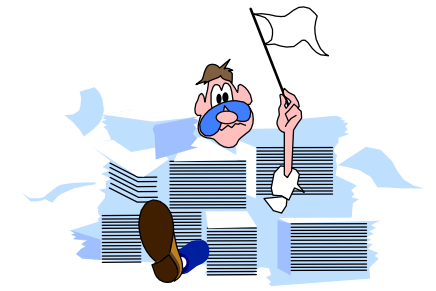
Data Base: STORE.DATA.INVENT  
By Robelle Consulting Ltd

Run on: MON, JAN 9, 1995, 11:48 AM  
Page: 1

Data Set	Type	Capacity	Entries	Load Factor	Secon- daries (Highwater)	Max Blks	Blk Fact
M-Customer	Man	248113	178018	71.7%	30.5%	1496	11
A-Order-No	Ato	1266783	768556	60.7%	25.7%	1	70
D-Orders	Det	1000000	768558	76.9%	( 851445)		32
D-Ord-Items	Det	4000000	3458511	86.5%	( 3470097)		23

Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong- ation
Customer-No	32	1.92	0.32	1.00	1.90	90.5%	1.90
Order-No	10	1.35	0.62	1.00	1.00	0.0%	1.00
!Order-No	1	1.00	0	1.00	1.00	0.0%	1.00
S Customer-No	80	14.34	17.76	1.75	9.20	57.2%	5.25
S !Order-No	1604	8.06	35.75	1.36	11.32	72.5%	8.34

# Interpreting master datasets lines



- Pay attention to the following statistics:
  - High percentage of Secondaries (inefficient hashing)
  - High Maximum Blocks (clustering)
  - High Maximum and Average Chains (inefficient hashing)
  - High Inefficient Pointers (when secondaries exist)
  - High Elongation (when secondaries exist)



# Report on m-customer

- The number of Secondaries is not unusually high
- However, there may be problems
  - Records are clustering (high Max Blks)
  - Long synonym chain
  - High percentage of Inefficient Pointers

Data Set	Type	Capacity	Entries	Load Factor	Secon- Max daries Blks (Highwater)	Blk Fact			
M-CUSTOMER	Man	248113	178018	71.7%	<u>30.5%</u> 1496	11			
	Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong- ation	
	CUSTOMER-NO	<u>22</u>	1.92	0.32	1.00	1.90	<u>90.5%</u>	1.90	

# Report on a-order-no



- Very tidy dataset
  - Number of Secondaries is acceptable
  - Max Blks, Ineff Ptrs and Elongation are at the minimum values, even if the maximum chain length is a bit high

Data Set	Type	Capacity	Entries	Load Factor	Secondaries (Highwater)	Max Blks	Blk Fact		
A-ORDER-NO	Ato	1266783	768556	60.7%	<u>25.7%</u>	<u>1</u>	70		
		Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elongation	
	Search Field	<u>10</u>	1.35	0.62	1.00	1.00	<u>0.0%</u>	<u>1.00</u>	

# Master dataset solutions



- Increase capacity to a higher odd number
- Increase the Blocking Factor
  - Increase block size
  - Reduce record size
- Change binary keys to type X, U, Z, or P
- Check your database early in the design
- Use HowMessy on test databases

# HowMessy Exercise 1



Data Set	Type	Capacity	Entries	Load Factor	Secon-Max daries Blks (Highwater)	Blk Fact		
A-MASTER	Ato	14505679	9709758	66.9%	36.8% 2395	29		
	Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong-ation
	MASTER-KEY	37	1.58	1.26	1.00	1.88	48.5%	1.88



# HowMessy sample report (detail dataset)



HowMessy/XL (Version 2.2.1)  
for TurboIMAGE/3000 databases

Data Base: STORE.DATA.INVENT  
By Robelle Consulting Ltd.

Run on: MON, JAN 9, 1995, 11:48 AM  
Page: 1

Data Set	Type	Capacity	Entries	Load Factor	Secon- daries (Highwater)	Max Blks	Blk Fact
M-CUSTOMER	Man	248113	178018	71.7%	30.5%	1496	1
A-ORDER-NO	Ato	126673	768556	60.7%	25.7%	1	70
D-ORDERS	Det	1000000	768556	76.9%	( 851445)		12
D-ORD-ITEMS	Det	4000000	3458511	86.5%	( 3470097)		23

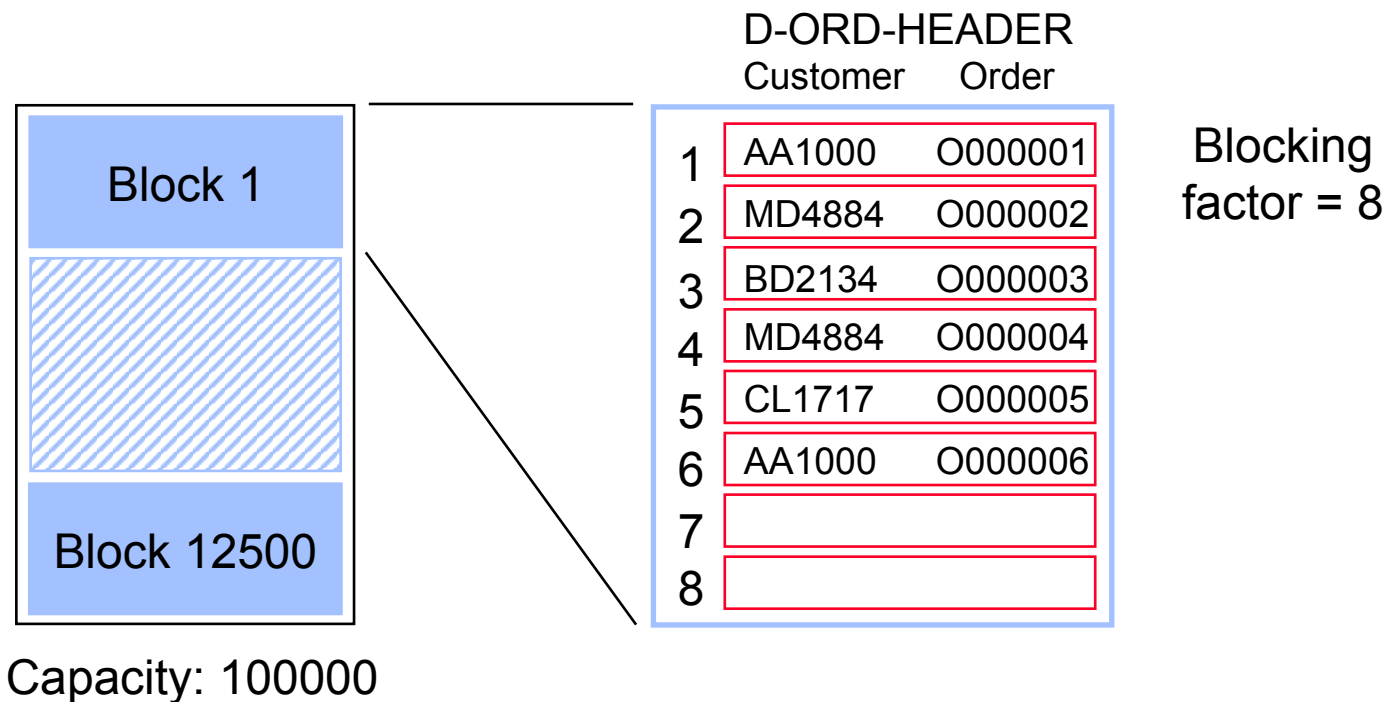
  

Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong-ation
Customer-No	22	1.92	0.32	1.00	1.90	90.5%	1.90
Order-No	10	1.35	0.62	1.00	1.00	0.0%	1.00
!Order-No	1	1.00	0	1.00	1.00	0.0%	1.00
S Customer-No	80	14.34	17.76	1.75	9.20	57.2%	5.25
S !Order-No	1604	8.06	35.75	1.36	11.32	72.5%	8.34



# Empty detail dataset

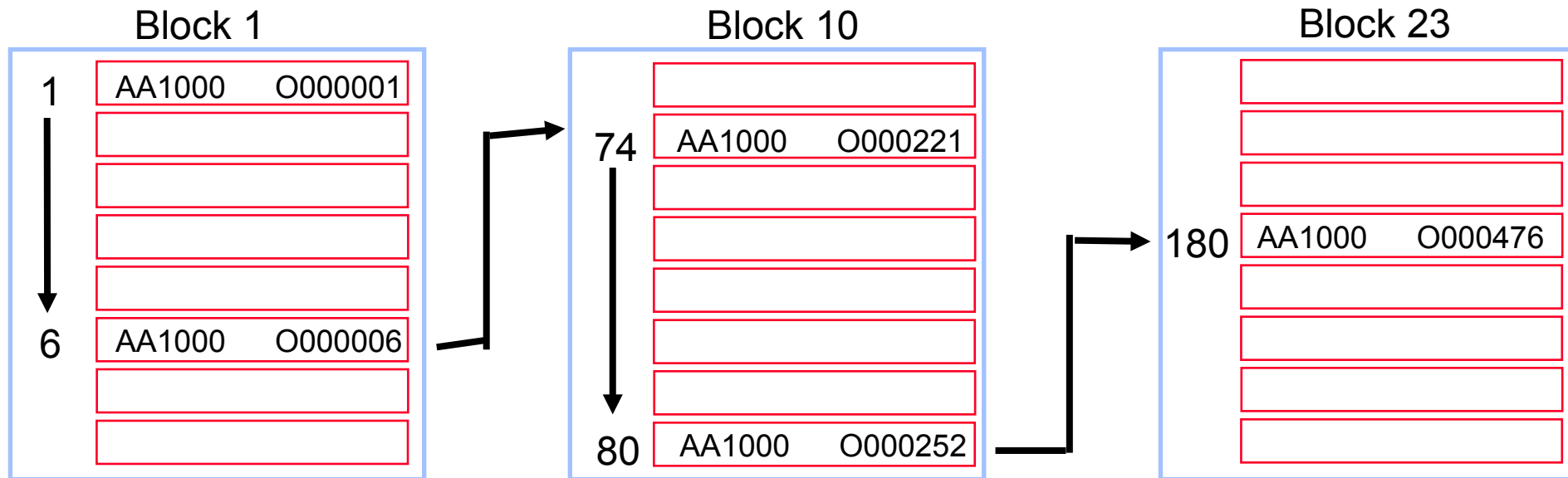
- Records are stored in the order they are created starting from record 1
- Records for the same customer are linked together using pointers to form a chain
- Chains are linked to the corresponding master entry





# Detail chains get scattered

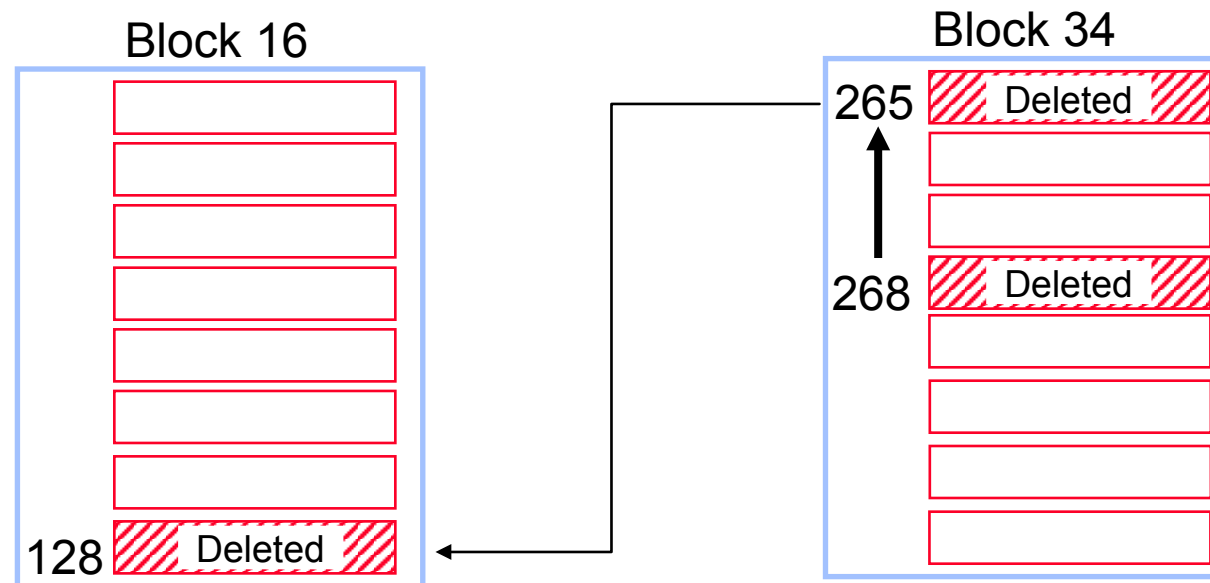
- Over time, records for the same customer are scattered over multiple blocks





# Delete chain

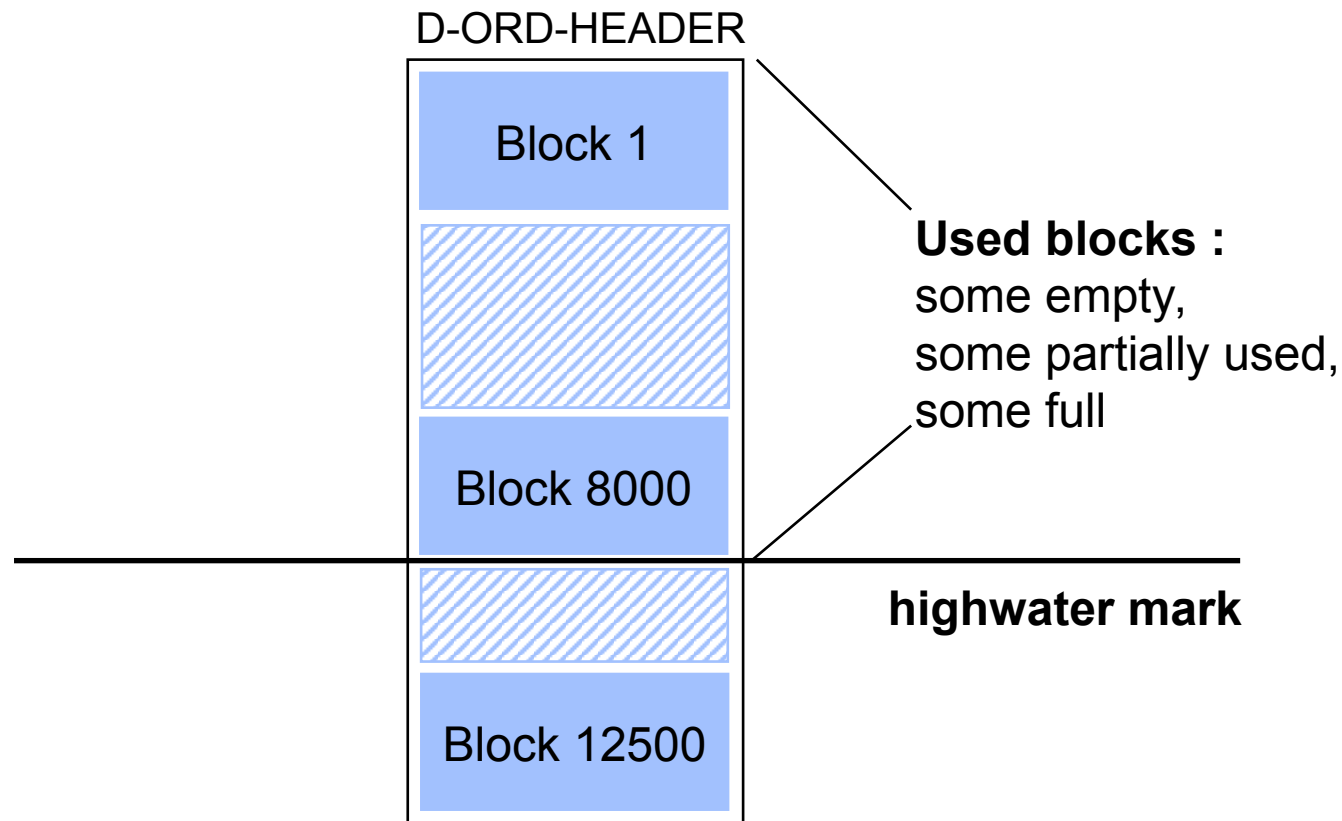
- Deleted records are linked together
- TurboIMAGE reuses the records in the Delete chain, if there are any





# Highwater mark

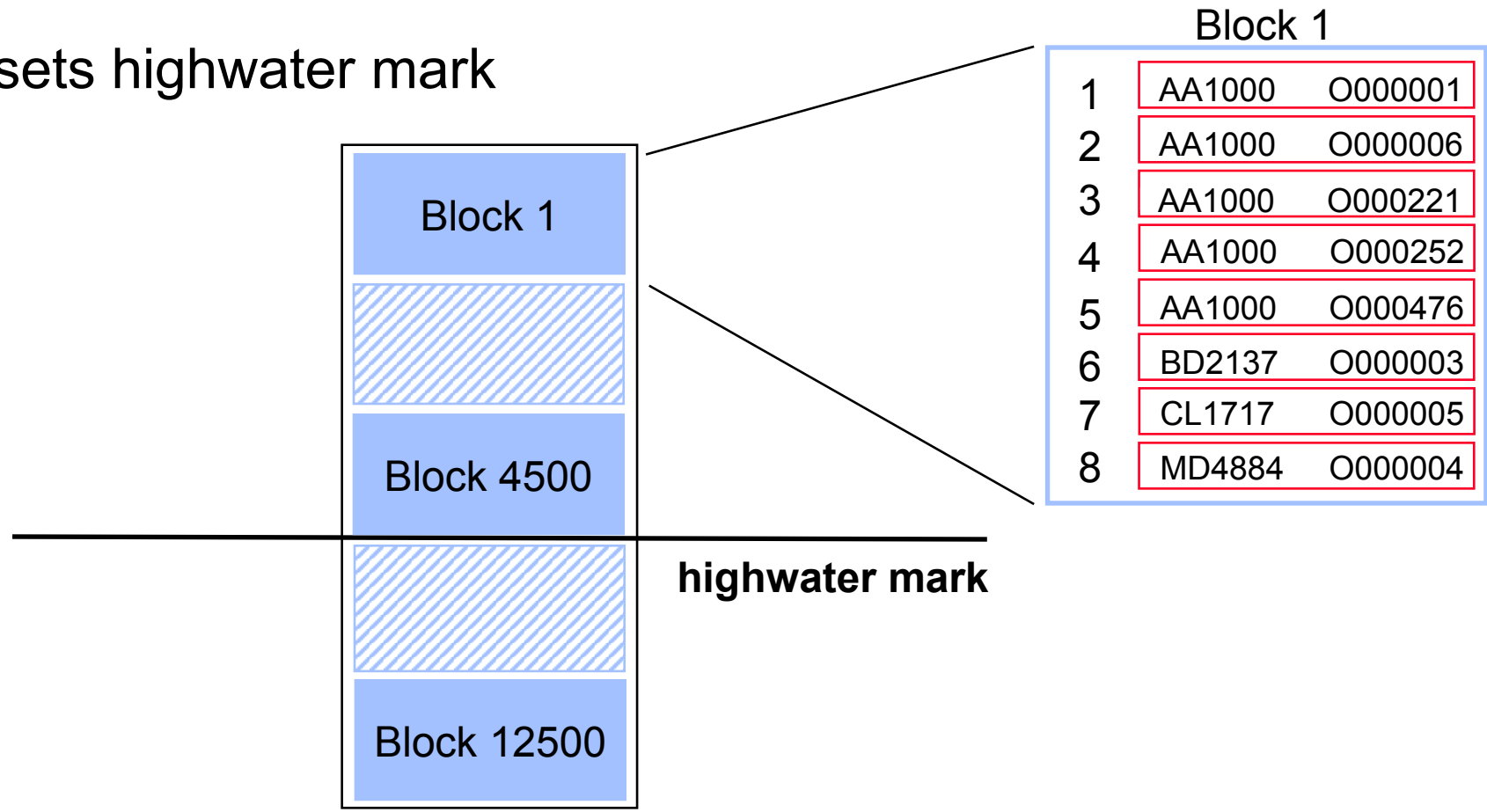
- Indicates highest record location used so far
- Serial reads scan the dataset up to the highwater mark



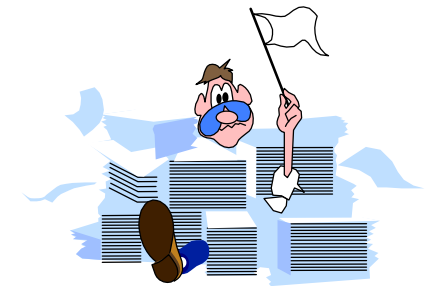


# Repacking a detail dataset

- Groups records along primary path
- Removes Delete chain (no holes)
- Resets highwater mark



# Interpreting detail dataset lines



- Pay attention to the following statistics:
  - Load Factor approaching 100% (dataset full)
  - Primary path (large Average Chain and often accessed)
  - High Average Chain and low Standard deviation, especially with a sorted path (Is path really needed?)
  - High Inefficient Pointers (entries in chain not consecutive)
  - High Elongation (entries in chain not consecutive)



# Report on d-orders

- Primary path should be on customer-no, not on order-no
- Highwater mark is high
- Repack along new primary path regularly

Data Set	Type	Capacity	Entries	Load Factor	Secon-Max daries Blks (Highwater)	Blk Fact		
D-ORDERS	Det	1000000	768556	76.9%	( <u>851445</u> )	12		
		Max	Ave	Std	Expd	Avg	Ineff	Elong-
	Search Field	Chain	Chain	Dev	Blocks	Blocks	Ptrs	ation
	!ORDER-NO	1	1.00	0	1.00	1.00	0.0%	1.00
	S CUSTOMER-NO	<u>80</u>	<u>14.34</u>	17.76	1.75	9.20	<u>57.2%</u>	5.25





# Report on d-ord-items

- Inefficient Pointers and Elongation are high
- Highwater mark is fairly high
- Repack the dataset regularly
- Is the sorted path really needed?

Data Set	Type	Capacity	Entries	Load Factor	Secon- daries Blks (Highwater)	Max Blks	Blk Fact	
D-ORD-ITEMS	Det	4000000	3458511	86.5%	( <u>3470097</u> )		23	
	Search Field	Max Chain	Ave Chain	Std Dev	Expd Blocks	Avg Blocks	Ineff Ptrs	Elong- ation
	S !ORDER-NO	1604	8.06	35.75	1.36	<u>11.32</u>	<u>72.5</u>	8.34

# Detail dataset solutions



- Assign the primary path correctly; search item with Average Chain length  $> 1$  that is accessed most often
- Repack datasets along the primary path regularly
- Increase the Blocking Factor
  - Increase block size
  - Reduce record size
- Understand sorted paths
- Check your databases early in the design; use HowMessy on test databases





# Minimum number of disc I/Os

## Intrinsic Disc I/Os

DBGET	1	
DBFIND	1	
DBBEGIN	1	
DBEND	1	
DBUPDATE	1	(non-critical item)
DBUPDATE	13	(critical item)
DBPUT	3	[+ (4 x #paths, if detail)]
DBDELETE	2	[+ (4 x #paths, if detail)]

Serial reads:

Master

Capacity / Blocking factor

Detail# entries / Blocking factor

# Estimating response time



- Deleting 100,000 records from a detail dataset with two paths would take:
  - $2 + (4 \times 2 \text{ paths}) = 10 \text{ I/Os per record}$
  - $100,000 \text{ records} \times 10 \text{ I/Os per record} = 1,000,000 \text{ I/Os}$
- Classic: around 25 I/Os per second
  - $1,000,000 \text{ I/Os} / 25 = 40,000 \text{ seconds}$
  - $40,000 \text{ seconds} / 3600 = 11.1 \text{ hours}$
- iX: around 40 I/Os per second
  - $1,000,000 \text{ I/Os} / 40 = 25,000 \text{ seconds}$
  - $25,000 \text{ seconds} / 3600 = 6.9 \text{ hours}$

# *Automating HowMessy analysis*



- Recent version of HowMessy creates a self-describing file with these statistics
- Process the file with generic tools (Suprtool, AskPlus) or custom programs (COBOL, 4GL), and produce custom reports
- Send messages to database administrators
- Write “smart” job to fix databases without user intervention

# Processing Loadfile with Suprtool



- Datasets more than 80% full

```
>input      loadfile
>if loadfactor > 80
>ext database, dataset, datasettype, loadfactor
>list standard
```

- Only one address per customer

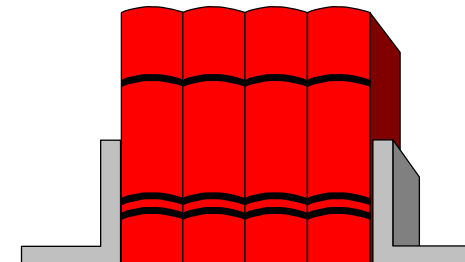
```
>input      loadfile
>if dataset = "D-ADDRESSES" and &
maxchain > 1
```

# References

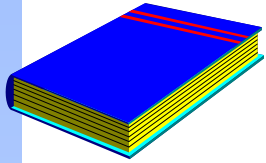


- The TurboIMAGE/3000 Handbook (Chapter 23)
- Available for \$ 49.95 from:

WORDWARE  
P.O. Box 14300  
Seattle, WA 98114







# Summary



- TurboIMAGE databases become messy over time, especially if they are active
- HowMessy and DBLOADNG let you analyze the database's efficiency
- You should have some knowledge of the internal workings of TurboIMAGE
- Monitor your databases regularly

