DB2 LOBs at work@ KBC

GSE 21/03/2013

Dirk Beauson
KBC Global Services

Agenda

- Business case 1
- LOB study
- Business case 2
- Business case 3
- Design lessons learned
- Feeding LOB data
  - Batch
  - Online
- LOB types used @ KBC

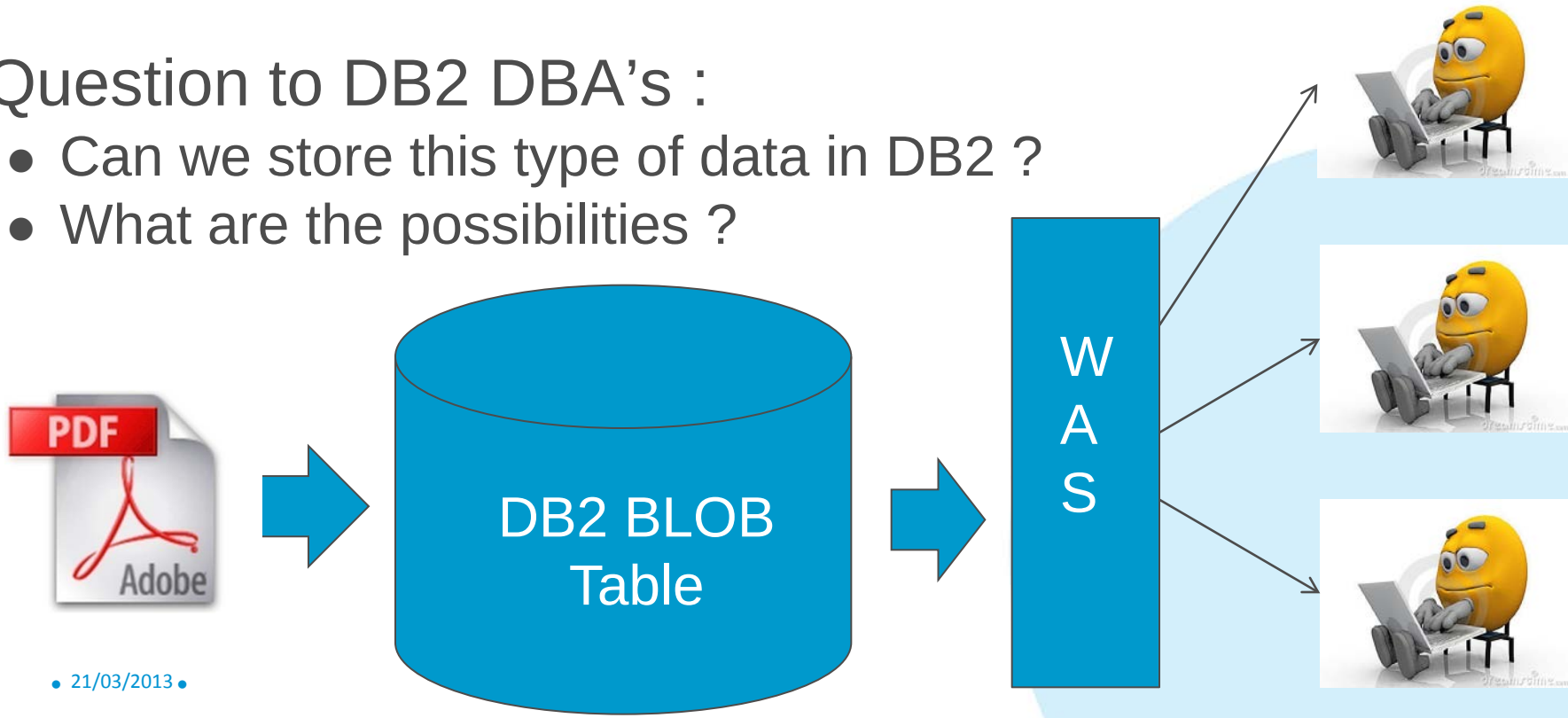
Agenda

- UNLOAD/LOAD
- DELETE vs Dummy LOAD
  - Design
- RECOVERY
- Problems during a release
- Problems with LOBs
- FUTURE

Business case 1

- In 2004 a business party wanted to show PDF files to our customers via KBC-Online

- Question to DB2 DBA's :
  - Can we store this type of data in DB2 ?
  - What are the possibilities ?



LOB study

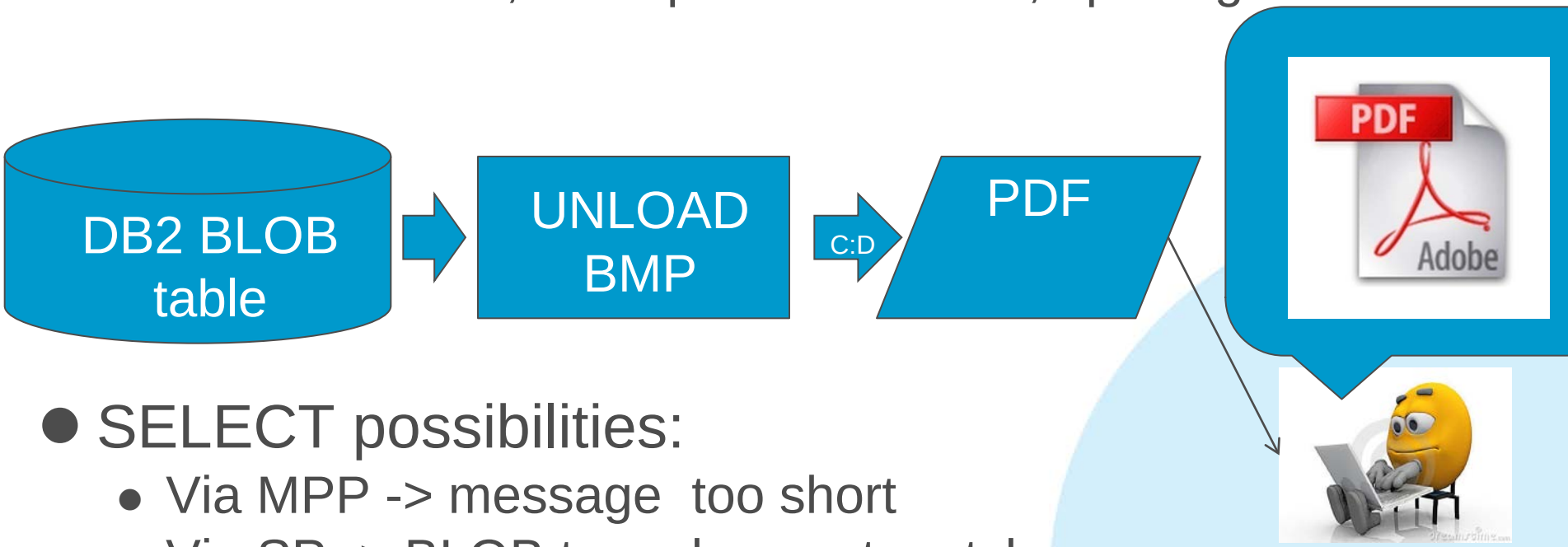
- Definition (DDL)
  - With Auxiliary objects
- Lots of data to store
  - Partitioning
- Std insert
- Std select
- Utilities :
  - RUNSTATS
  - REORG
    - Not really freeing space
  - IC

LOB study

- PDF file size :
  - Average : 150 KB
  - Max : 3 MB
- Some utilities only can handle LOBs up to 32K
- UNLOAD/LOAD will have to be done in another way :
  - BMP
    - Upload PDF to flat file on MF
    - Read it into memory or work with locators
    - Insert it

LOB study

- Test :
  - What will the PDF look like when we read it via a BMP, store it in a file, and upload it to a PC, opening the PDF



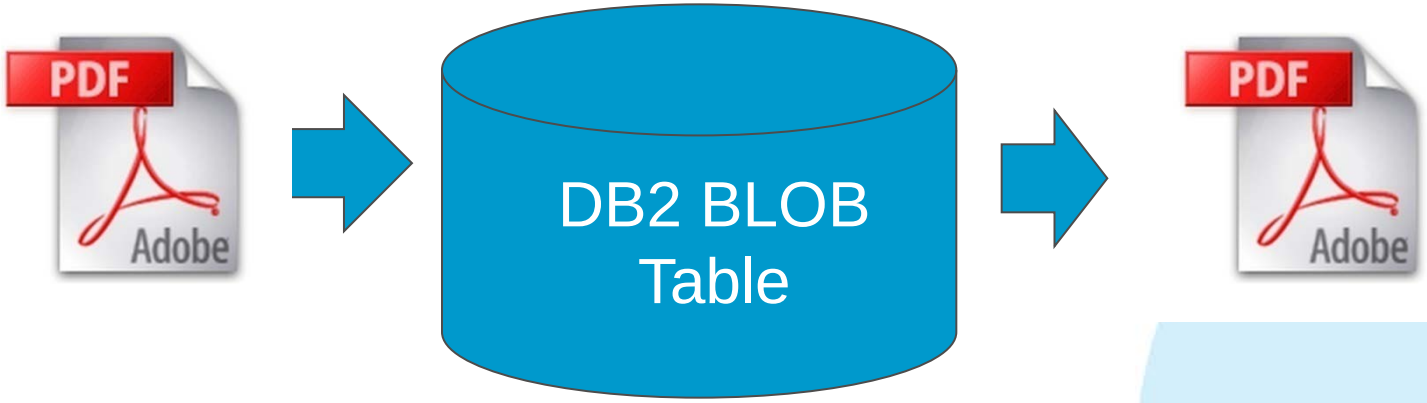
- SELECT possibilities:
  - Via MPP -> message too short
  - Via SP -> BLOB type does not match
  - Via Dyn SQL -> only possibility left via JDBC -> ok

LOB study

- Other question :
  - How can we upload multiple PDF files -> MF and insert them all at once
  - Talked to the Connect:Direct, MQ guys
  - They can group all documents in 1 file
  - The business can code a separator upfront of the PDF containing a unique key
  - Layout :
    - A separator
    - Unique key
    - A separator
    - A PDF (y lines in the file)
    - A separator with unique key
    - A PDF
    - ...

LOB Study

- Format to unload and load BLOBs > 32K via BMP
- Inserted LOBS can be read
- Result is the same PDF file as inserted.



- Everything seems to work so let’s go for it

TB design
Business case 1

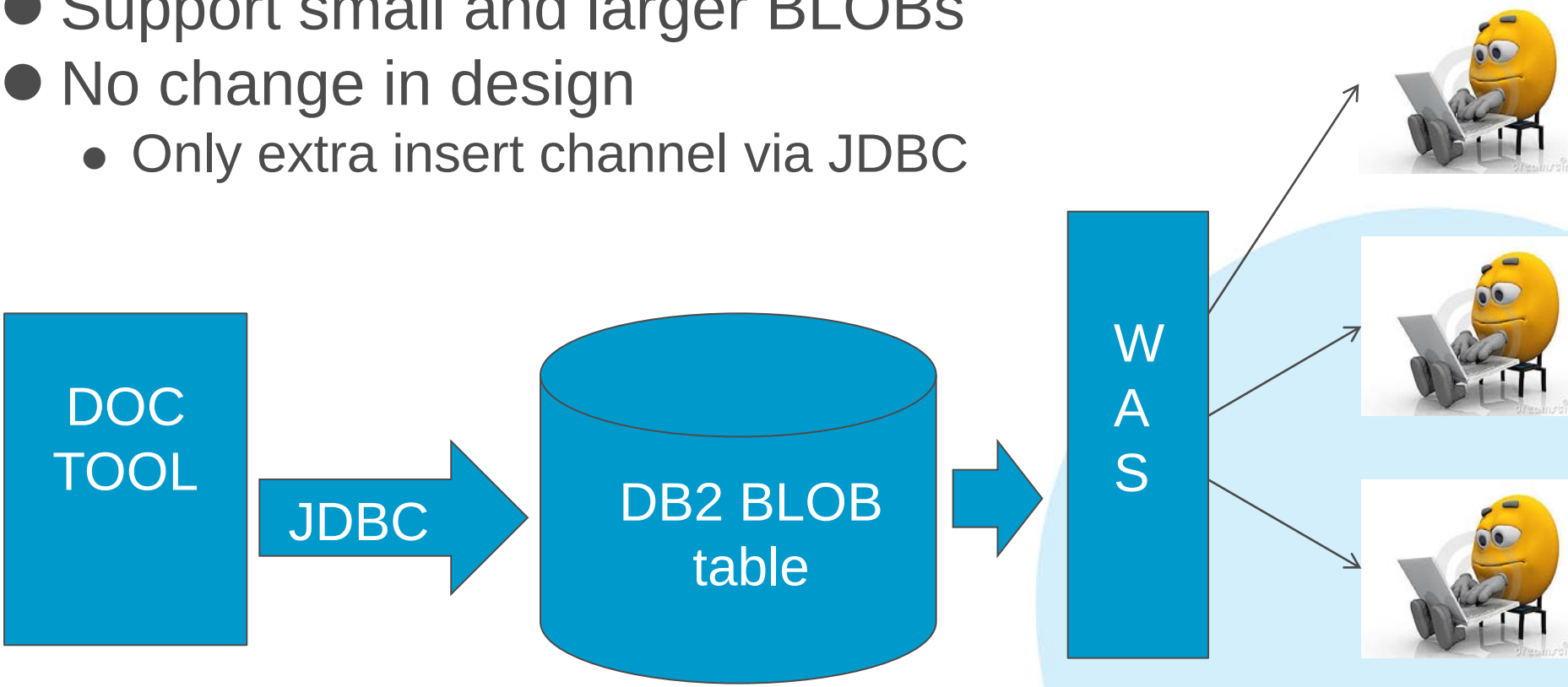
- A lot of small BLOBs :
  - 50 à 60 KB
- And larger BLOBs :
  - 100 à 200 KB
  - MAX 3M
- Advise IBM :
  - Design separate partitions for small and larger LOB objects
  - Better reuse of free space

TB design
Business case 1

- RESULT :
  - 20 pt small / 10 pt larger
    - >= 100 KB
- 4G per partition
  - No partion >4 G
  - No need for extra datasets
- DELETE process dailly via BMP
- Everything was set up as described AND it worked

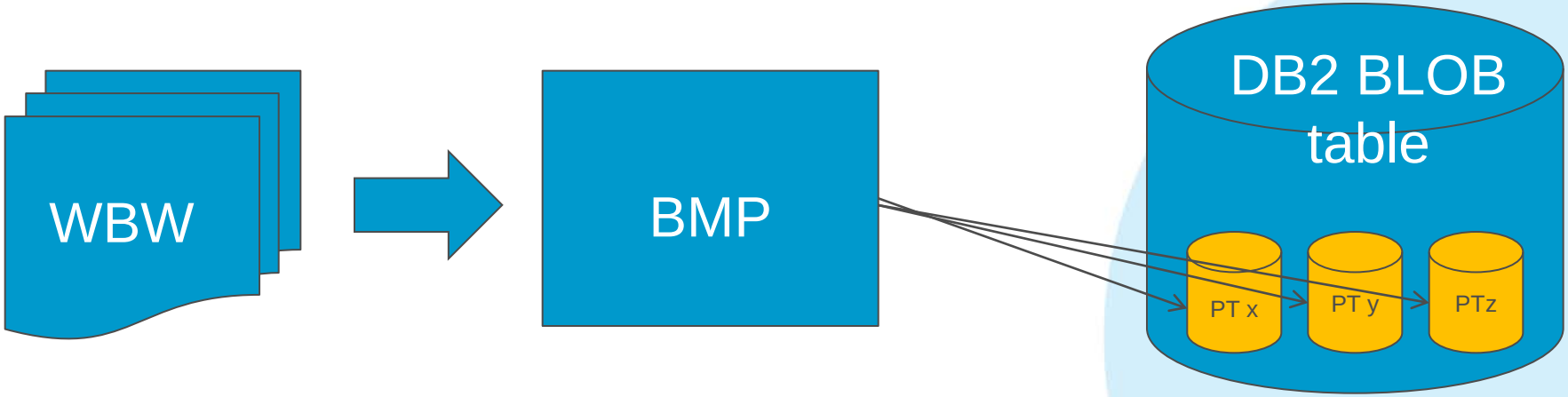
Extra functionality 1
Business case 1

- INSERT from a remote document tool
- Support small and larger BLOBs
- No change in design
  - Only extra insert channel via JDBC



Extra functionality 2
Business case 1

- Insert a few million files in january into the current BLOB-table design
- Selectable via KBC online
- Delete at the end of march



Extra functionality 2
Business case 1

- Didn’t fit in our already implemented BLOB design :
  - To much data inserted to LOG
  - Same problem -> delete
  - Add 3 extra LOB partitions LOG NO
  - Loadfile = our backup
  - INSERT via BMP per partition
  - DELETE via Dummy Load partition
- Disadvantage :
  - Rest of the year partitions empty
  - Reusable for other processes in another period

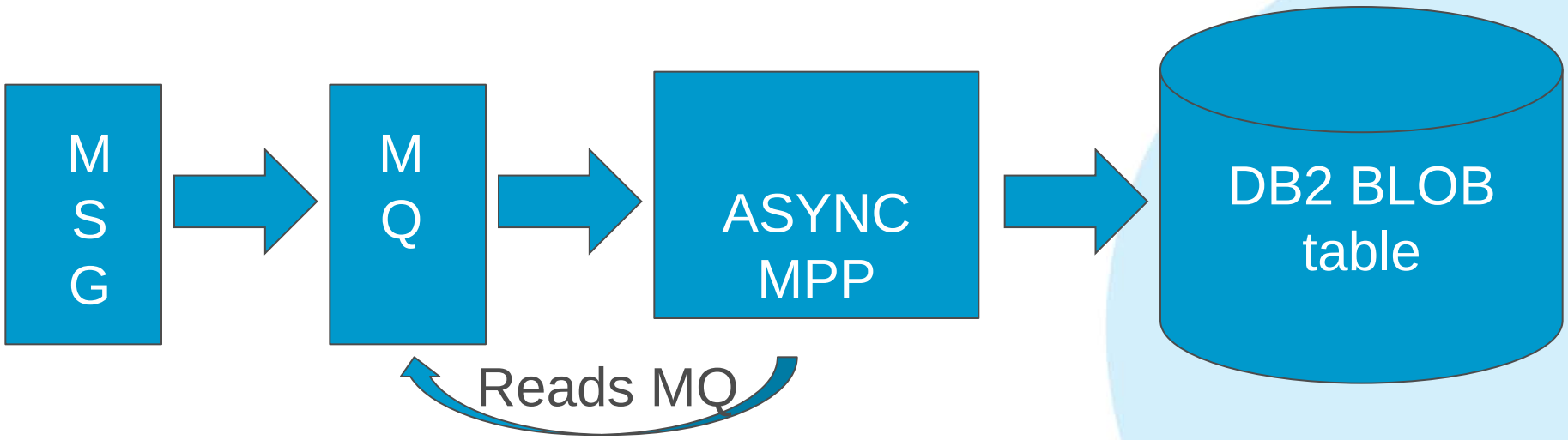
Daily behaviour business case 1

online		batch	
insert	select	insert	delete
30000	40000	10000	40000

Number of rows on table : +- 850.000

Business case 2

- Store a signature of a special kind of payment
- Small BLOB : +- 10 KB
- Just INSERT data via MQ -> ASYNC MPP
- Afterwards after x days, DELETE data
- Max 100.000 rows in table



Daily behaviour business case 2

online		batch	
insert	select	insert	delete
2000	0	30000	32000

Number of rows on table : +- 50.000 à 100.000

TB design
Business case 2

- Only 1 partition needed
- 4G

- Extra functionality :
  - INSERT daily via BATCH

Business case 3

- Store
  - whatever document
  - for max 13 months
  - with a manageable volume / month (max 32 G) / category
- A category can be :
  - BATCH
  - ONLINE
  - Specific kind of data

Business case 3

- To be recoverable at any time
  - Batch load can be up to many G at a time
  - Limit system guys logging per batch run : per 2 G
  - -> Large Batch runs LOG NO !!!
  - Load file = BU
- Only 1 table allowed -> less CREATE COST

Some figures Business case 3

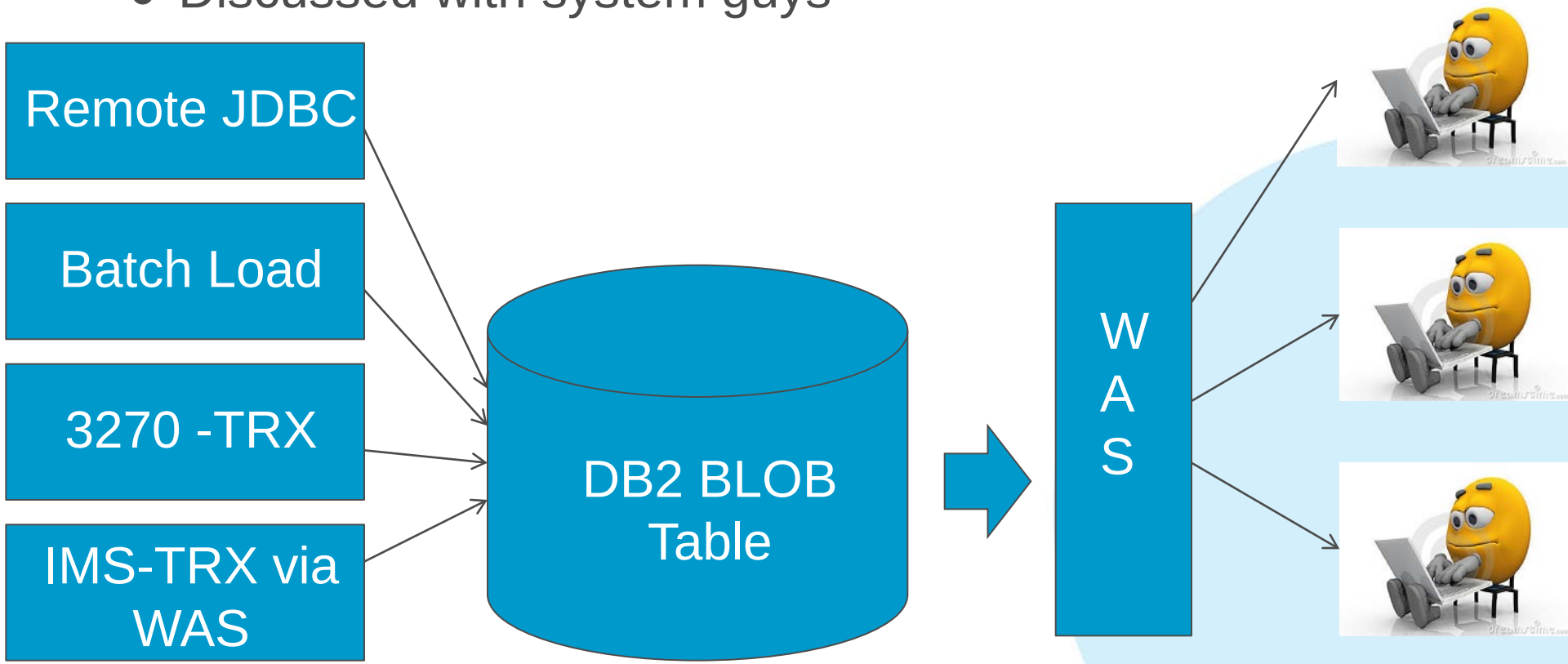
Soort	Aantal berichten/dag	Aantal berichten/jaar	Gem. grootte bericht (Kb)	Gb blob na 1 jaar	Gedrag	Frequentie	Piek
workload 1	20000	7300000	28	194,93	Online	dagelijks	Nee
workload 2	50000	7000000	3	20,03	Batch/Online	dagelijks	Nee
workload 3	50000	13700000	3	39,20	Batch/Online	dagelijks	**ja, tot 1000000 per dag !!!**
workload 4	50000	14023277	22	294,22	???	dagelijks	**ja, tot 1000000 per dag !!!**
workload 5	/	1000000	5	4,77	Batch	jaarlijks	1000000 -> 5GB (5Kb per msg)
workload 6	/	1000000	5	4,77	Batch	jaarlijks	1000000 -> 5GB (5Kb per msg)
workload 7	/	1000000	22	20,98	Batch	jaarlijks	**gespreid over 2, 3 en 4 oktober**
workload 8	2000	500000	50	23,84	Batch	dagelijks	Nee
workload 9	160	40000	20	0,76	Batch	dagelijks	Nee
workload 10	200	50000	150	7,15	Batch	dagelijks	Nee
workload 11	400	100000	120	11,44	Batch	dagelijks	Nee
workload 12	100	25000	20	0,48	Batch	dagelijks	Nee
workload 13	88	22000	100	2,10	Batch	dagelijks	Nee
workload 14	36	9000	23	0,20	Batch	dagelijks	Nee
workload 15	0	0	15	0,00	Batch	dagelijks	Nee
workload 16	0	0	65,86	0,00	Batch	dagelijks	Nee
workload 17	5,2	1300	41	0,05	Batch	dagelijks	Nee
workload 18	44	11000	55,36	0,58	Batch	dagelijks	Nee
workload 19		600000	23	13,16	Batch	maandelijks rond de 20ste	maandelijks 50000 (=op 1 dag!!) -> **1 Gb**
workload 20	29,04	7260	22	0,15	Batch	dagelijks	Nee
workload 21	200	50000	55	2,62	Batch	dagelijks	Nee
workload 22		400000	11	4,20	Batch	maandelijks 1e werkdag	**200000 (januari, op 1 dag!!!!!) -> 2 Gb**
workload 23		480000	90	41,20	Batch	maandelijks	40000 op 1 dag
workload 24		192000	10	1,83	Batch	maandelijks	16000 op 1 dag, rond de 10e
workload 25		375000	150	53,64	Batch	jaarlijks	februari 1 dag 175000 -> **25GB** + nog 3 dagen (eind januari/beginfebruari)
workload 26		24000	150	3,43	Batch	??	??
workload 27		36000	150	5,15	Batch	maandelijks	?
workload 28	?	2000000	150	286,10	Batch	dagelijks	jaarlijks piek in februari: 1800000, gespreid over een aantal dagen

TB design
Business case 3

- Analysis:
  - 11 patterns found
  - 6 * 13 (months) partitions
  - 3 * 1 (month) partition
  - 1 * 2 (months) partitions
  - 1 * 10 (months) partitions
  - 37 free partitions
  - Total : 130 partitions
- For heavy batch load -> LOG NO
- 1 partition per month
  - DELETE = dummy load partition
  - Reduce logging !!!

TB design
Business case 3

- 8G DSSIZE
  - Max 4 datasets per partition
  - Discussed with system guys

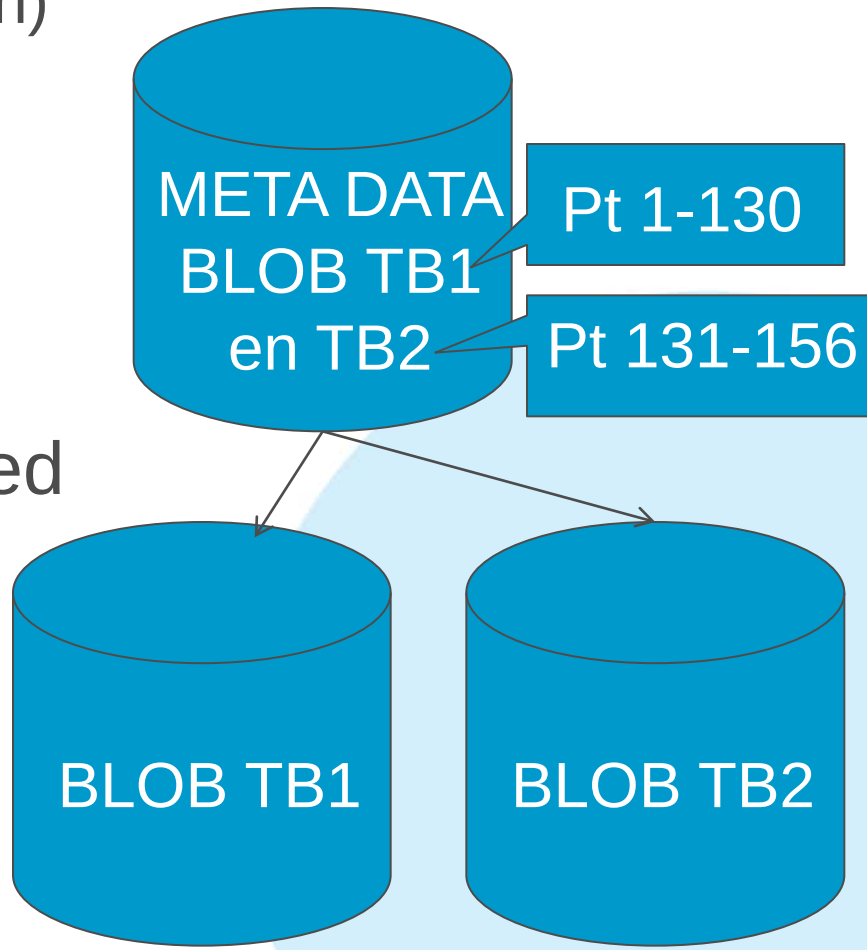


Extra functionality
Business case 3

- NO limited volumes / month
- Only batch load
- NO limit (first 64 G)
- Not to be recoverable
  - Cleanup
  - Rerun process -> ok
- Completely different pattern
  - Not suitable in current design
  - Extra LOB table designed

TB design Extra functionality Business case 3

- Extra table added with BLOB column
  - Also 13 partitions (1 per month)
  - 8G DSSIZE
  - LOG NO
  - Load file = BU
  - Or process can be restarted again after cleanup
- In this case Meta Data shared
  -> less build cost
  -> but not optimal
  (Future PT adding)



Daily behaviour business case 3

online		batch	
insert	select	insert	delete
1300	25000	500	0

Number of rows on table : +- 17.500.000

Design lessons learned

- Putting as much as possible in 1 LOB table isn't a good idea
- Maintenance of all LOB partitions has to be guaranteed
  - Have to be manageable parts !!!
- Separate meta data from LOB data
  - Meta data
  - LOB table
    - Key
    - LOB column
    - No need to ever change table layout

Design lessons learned

- Put LOB data where it belongs
  - Persons LOB data belongs in het persons system
  - Smaller manageable kind of data
  - Can still be a large amount of data
- LOB Meta data :
  - Can be put together if possible
  - No necessity
- Mass deletion
  - Pay attention -> LOG
  - Dummy load partition
    - Reduce logging !

New Designes

- LOB data where it belongs
- Including the Meta data
- Smaller systems
  - Manageable
  - Maintainable
  - …

Feeding LOB data

- Ways of loading LOB data @ KBC :
  - BATCH <= 2G
    - Input file with separators and keys
    - INSERT via BMP :
      - ¬ At once if possible in BMP (memory allocation)
      - ¬ Via locators and update
    - LOG YES
  - BATCH > 2G
    - Input file with separators and keys
    - INSERT via BMP :
      - ¬ At once if possible in BMP
      - ¬ Via locators and update
    - LOG NO
    - Inputfile = BU !!!
    - In some cases daily unload = BU

Feeding LOB data

- ONLINE
  - INSERT from WAS/JAVA application via JDBC
    - ¬ LOG YES
  - Via MQ -> async MPP
    - ¬ INSERT in MPP
    - ¬ LOG YES
  - INSERT via 3270 TRX
    - ¬ Cast small text into binary data and insert
    - ¬ LOG YES

LOB types used @ KBC

- For the moment we only use :
  - BLOBs
- XML-study planned

UNLOAD/LOAD

- Because most of our LOBs > 32K
  - No use of :
    - UNLOAD
    - LOAD
- LOB design -> 3 BMP's :
  - One for unloading the data
  - One for loading the data
  - One for unloading the data from independent outspace

  - Layout:
    - Separator
    - Key
    - Lob data (x lines)
    - …

DELETE vs Dummy Load

- DELETE ->LOG
- Periodically delete process (once a month)
  - Special Design
  - No fancy stuff
  - Designing partitions
  - DELETE via :
    - Dummy Load LOG(NO)
    - inline copy (base table pt)
    - extra copy of the LOB TS (PT)
  - Application inserts in the partition defined to be deleted at the end of the lifecycle of the inserted BLOB.

DELETE vs Dummy Load

- Other delete processes :
  - Delete via BMP
  - Reorg on a regular base
    - Free space

RECOVERY via INDEPENDENT OUTSPACE

- Recover BLOB data
  - Never to the base TS/PT.
  - Always to an independent outspace
  - From that independent outspace
    - Needed BLOBs selected via a unload BMP on the independent outspace
  - Selection uploaded via the load BMP
  - The unload BMP of the independent outspace is a copy of the std unload BMP, only difference is the name of the table.

Problems during a release

- Changing a table with a BLOB column :
  - Takes too long
  - Using the BMPs
- Eliminate the chance of changing the LOB-table
  - Meta data in a separate table

Problems with LOBs

- Once a problem occurred with a LOB TS :
  - Simple table:
    - Key (char(20))
    - BLOB (3M) (avg length : 15K)
  - No partitions
  - DSSIZE 4G
  - TB full message
  - REORG didn’t release free space
  - No reuse of freespace
  - Even deletion of rows didn’t help
  - Normally DB2 should allocate an extra DS, but didn’t do it

Problems with LOBs

- Only thing that helped :
  - UNLOAD
  - DROP/CREATE
  - LOAD

- At that time there was a new algorithm for LOBs about freeing space
- Concurrency of multiple threads and hanging threads could impact reuse of space
- No hanging thread in DB2 on that LOB TB.

Problems with LOBs

- Problem opened @ IBM
- Problem recognized @ IBM
- FIX
- Problem solved
  - No more problems reusing freed space
  - DB2 takes an extra dataset, if needed

FUTURE

- Study DB2 9 and DB2 10 LOB features :
  - Automatic creation of LOB objects
  - Inline LOBS
  - BPSIZE/BUFFERPOOL
  - PAGESIZE
  - DSSIZE
  - GBPCACHE
  - UTILITIES
  - UNLOAD/LOAD VBS Spanned Record Format

Automatic creation of LOB objects

- CREATOR : SYSIBM
- DBNAME : DSN%
- Default parameter setttings

- Generated naming convention not allowed @ KBC

Inline LOBS

- Part of the LOB or whole LOB in base table
- Only for smal LOBs
  - Have to fit in 32K
- If LOB > size left in base table
  - First part in base table
  - Rest in aux LOB table
- Advantages :
  - Disk space usage (1 LOB/pg)
  - Compression
  - CPU savings

Inline LOBS

- Tests done
  - Insert lobs completely inline
  - Insert lobs partly inline
  - Select lobs completely inline
  - Select lobs not completely inline
- In most cases same or greater CPU usage :
  - Select
  - Insert
  - Delete
- Less GP (half) when inserting inline only, but more CPU ???
  - Other case less GP, but not that much

Inline LOBS

- When only insert inline, we see access to :
  - PPINFOTABLE ???
  - PPINFOTABLE_IX1 ???
- Usage @KBC :
  - Designs > 32 K BLOBs
  - No direct advantage
  - Keep option in mind

BPSIZE/BUFFERPOOL

- All our LOB TS -> Separate BP
- Low re-referencing -> BP not that big

PAGESIZE

- All our LOBS -> 4K pages
- We should look @ this :

Average LOB size (ALS)	Recommended page size for LOB table space
ALS < 4 KB	4 KB
4 KB < ALS < 8 KB	8 KB
8 KB < ALS < 16 KB	16 KB
16 KB < ALS	32 KB

DSSIZE

- Case by case
- With the system guys
- Used :
  - 4G
  - 8G

GBPCACHE

- All LOB TS -> GBPCACHE SYSTEM
  - Default in DB2 8
  - From DB2 9
    - GBPCACHE CHANGED can be considered, because of the changes in LOB locks management
  - Change ?

UTILITIES

- RUNSTATS
  - Schedule LOBs separately
- REORG SHRLEVEL REFERENCE
  - Free Space
- COPY
  - No inline copy (only base table space)
  - Shedule extra copy step on LOB TS

UNLOAD/LOAD VBS spanned record format

- Tests done :
  - UNLOAD/ LOAD : ok
  - UNLOAD/LOAD per partition : ok
  - UNLOAD/LOAD per partition all partitions : nok

- Remarks :
  - Dataset no EDIT/BROWSE

UNLOAD/LOAD VBS spanned record format

CBTXXX01	Via BMP (CPU)	Via BMP (Elapsed)	Via Utilitiy (CPU)	Via Utilitiy (Elapsed)	# rows	AVG BLOB Size
Unload	12:12.97	1:29:40	1:23.05 (-88%)	22:06	2682431	34M
Load	12:52.23	1:07:19	1:06.18 (-92%)	19:03		

CBTXXX02	Via BMP (CPU)	Via BMP (Elapsed)	Via Utilitiy (CPU)	Via Utilitiy (Elapsed)	# rows	AVG BLOB Size
Unload	8:32.63	2:06:04	1:26.76 (-83%)	1:33:08	800000	56M
Load	9:06.16	54:10:00	1:57.76 (-78%)	22:34		

UNLOAD/LOAD VBS spanned record format

- Planned to use during a release
  - Less CPU
  - But also less ELPTIME !!!