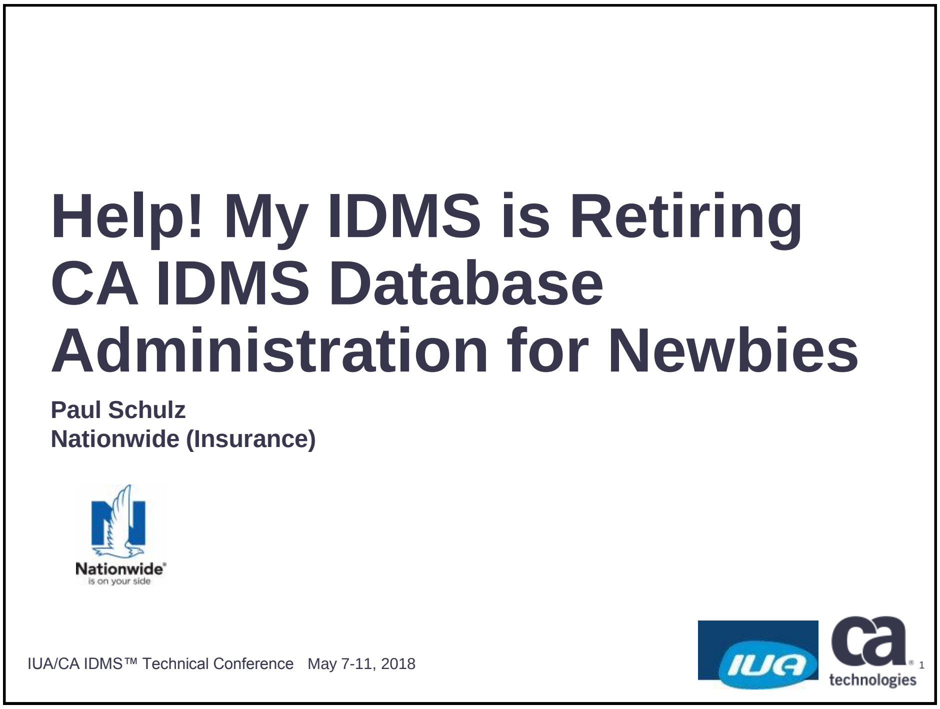
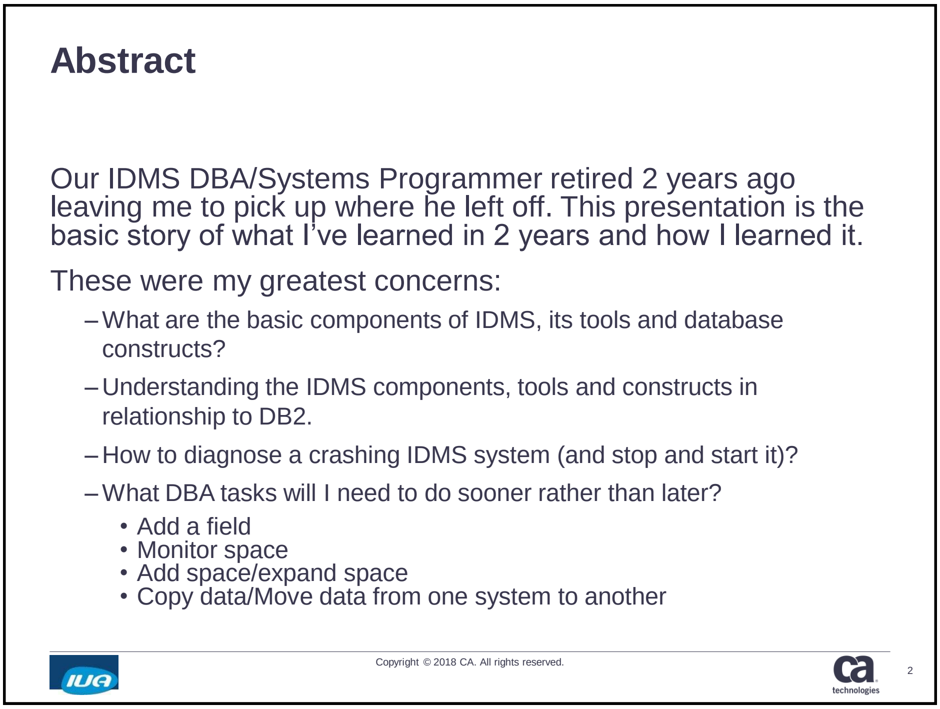


Help! My IDMS is Retiring CA IDMS Database Administration for Newbies

Paul Schulz
Nationwide (Insurance)

IUA/CA IDMS™ Technical Conference May 7-11, 2018

1



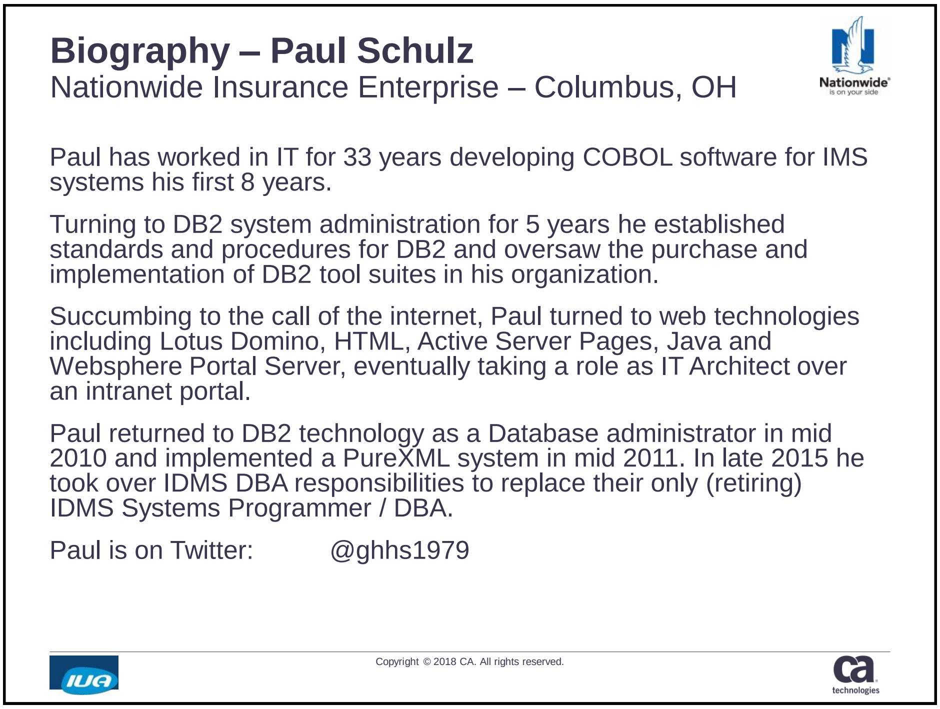
Abstract

Our IDMS DBA/Systems Programmer retired 2 years ago leaving me to pick up where he left off. This presentation is the basic story of what I've learned in 2 years and how I learned it.

These were my greatest concerns:

- What are the basic components of IDMS, its tools and database constructs?
- Understanding the IDMS components, tools and constructs in relationship to DB2.
- How to diagnose a crashing IDMS system (and stop and start it)?
- What DBA tasks will I need to do sooner rather than later?
  - Add a field
  - Monitor space
  - Add space/expand space
  - Copy data/Move data from one system to another

2



Biography – Paul Schulz

Nationwide Insurance Enterprise – Columbus, OH

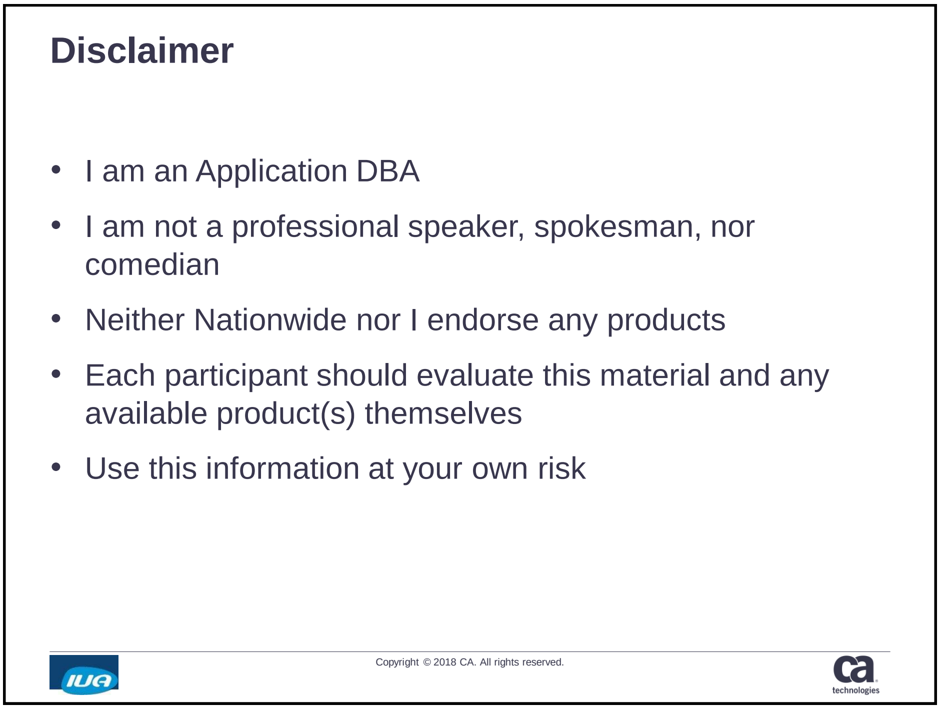
Paul has worked in IT for 33 years developing COBOL software for IMS systems his first 8 years.

Turning to DB2 system administration for 5 years he established standards and procedures for DB2 and oversaw the purchase and implementation of DB2 tool suites in his organization.

Succumbing to the call of the internet, Paul turned to web technologies including Lotus Domino, HTML, Active Server Pages, Java and Websphere Portal Server, eventually taking a role as IT Architect over an intranet portal.

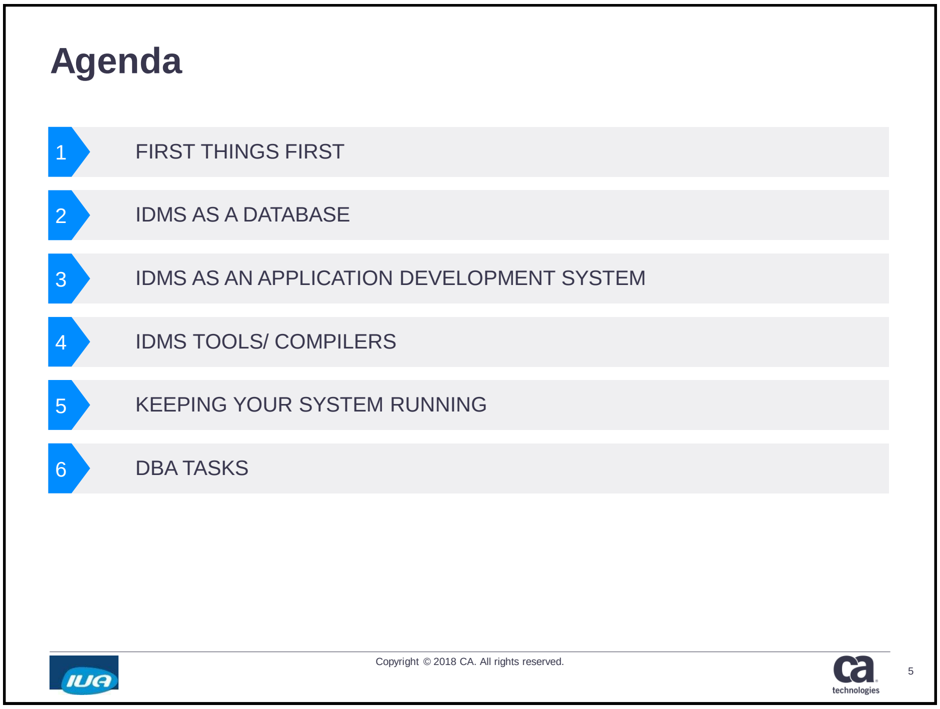
Paul returned to DB2 technology as a Database administrator in mid 2010 and implemented a PureXML system in mid 2011. In late 2015 he took over IDMS DBA responsibilities to replace their only (retiring) IDMS Systems Programmer / DBA.

Paul is on Twitter: @ghhs1979



Disclaimer

- I am an Application DBA
- I am not a professional speaker, spokesman, nor comedian
- Neither Nationwide nor I endorse any products
- Each participant should evaluate this material and any available product(s) themselves
- Use this information at your own risk



Agenda

1 FIRST THINGS FIRST

2 IDMS AS A DATABASE

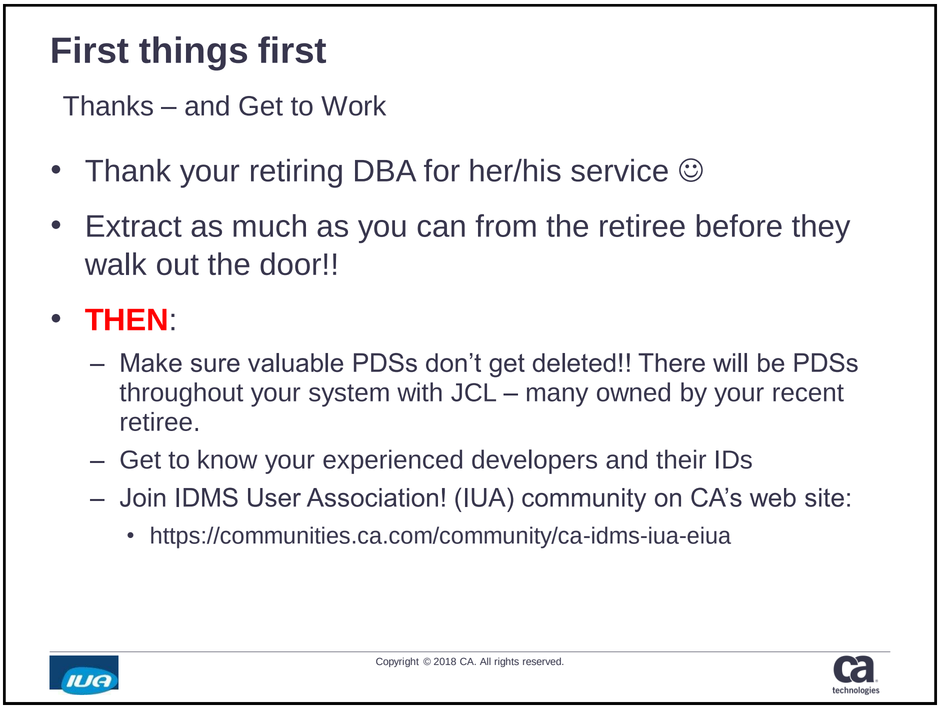
3 IDMS AS AN APPLICATION DEVELOPMENT SYSTEM

4 IDMS TOOLS/ COMPILERS

5 KEEPING YOUR SYSTEM RUNNING

6 DBA TASKS

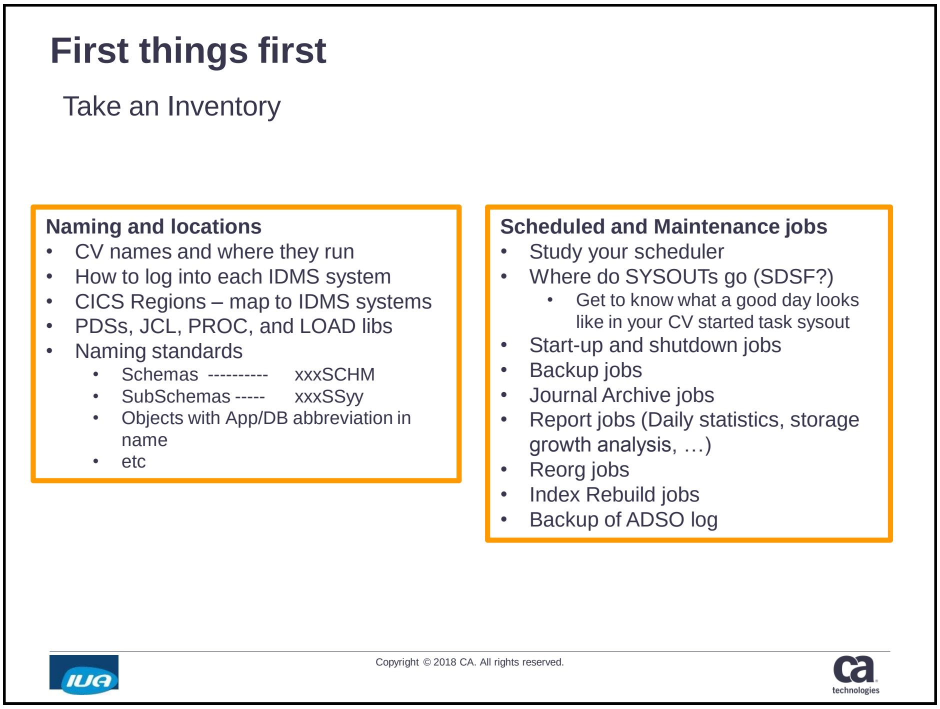
First things first



First things first

Thanks – and Get to Work

- Thank your retiring DBA for her/his service ☺
- Extract as much as you can from the retiree before they walk out the door!!
- **THEN**:
  - – Make sure valuable PDSs don't get deleted!! There will be PDSs throughout your system with JCL – many owned by your recent retiree.
  - – Get to know your experienced developers and their IDs
  - – Join IDMS User Association! (IUA) community on CA's web site:
    - https://communities.ca.com/community/ca-idms-iua-eiua



First things first

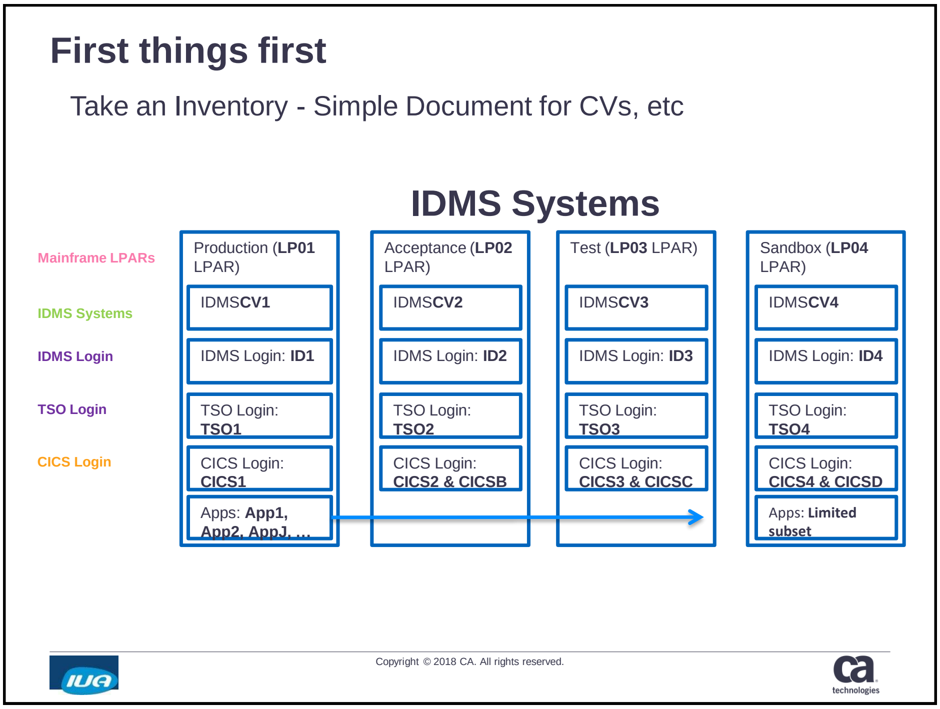
Take an Inventory

Naming and locations

- CV names and where they run
- How to log into each IDMS system
- CICS Regions – map to IDMS systems
- PDSs, JCL, PROC, and LOAD libs
- Naming standards
  - Schemas ---------- xxxSCHM
  - SubSchemas ----- xxxSSyy
  - Objects with App/DB abbreviation in name
  - etc

Scheduled and Maintenance jobs

- Study your scheduler
- Where do SYSOUTs go (SDSF?)
  - Get to know what a good day looks like in your CV started task sysout
- Start-up and shutdown jobs
- Backup jobs
- Journal Archive jobs
- Report jobs (Daily statistics, storage growth analysis, …)
- Reorg jobs
- Index Rebuild jobs
- Backup of ADSO log

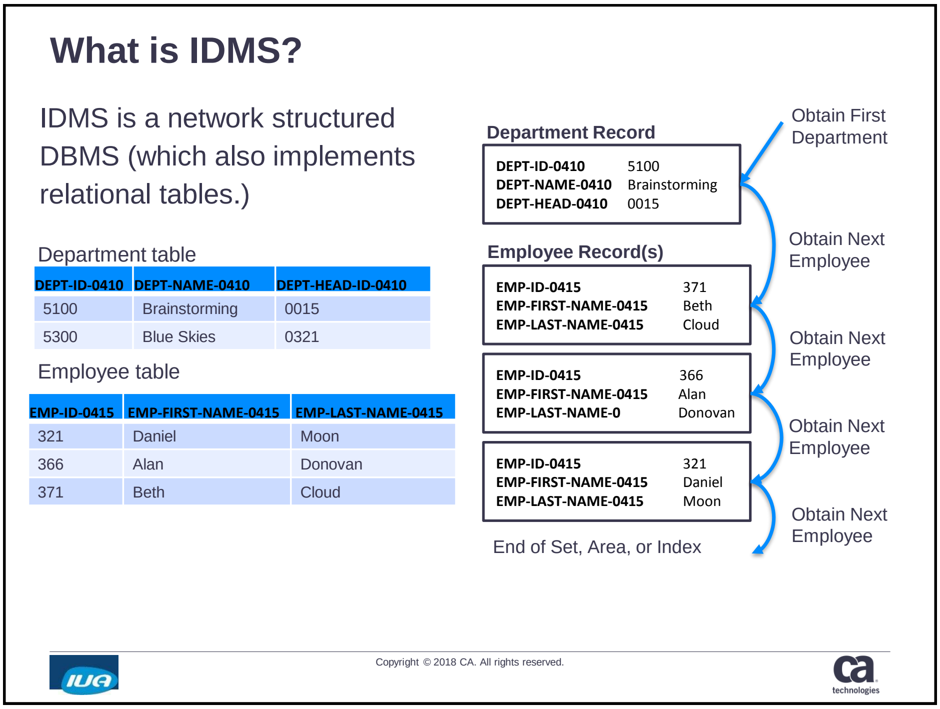


First things first

Take an Inventory - Simple Document for CVs, etc

IDMS Systems

Mainframe LPARs	Production (**LP01** LPAR)	Acceptance (**LP02** LPAR)	Test (**LP03** LPAR)	Sandbox (**LP04** LPAR)
IDMS Systems	IDMS**CV1**	IDMS**CV2**	IDMS**CV3**	IDMS**CV4**
IDMS Login	IDMS Login: **ID1**	IDMS Login: **ID2**	IDMS Login: **ID3**	IDMS Login: **ID4**
TSO Login	TSO Login: **TSO1**	TSO Login: **TSO2**	TSO Login: **TSO3**	TSO Login: **TSO4**
CICS Login	CICS Login: **CICS1**	CICS Login: **CICS2 & CICSB**	CICS Login: **CICS3 & CICSC**	CICS Login: **CICS4 & CICSD**
	Apps: **App1, App2, AppJ, …**			Apps: **Limited subset**



What is IDMS?

IDMS is a network structured DBMS (which also implements relational tables.)

Department table

DEPT-ID-0410	DEPT-NAME-0410	DEPT-HEAD-ID-0410
5100	Brainstorming	0015
5300	Blue Skies	0321

Employee table

EMP-ID-0415	EMP-FIRST-NAME-0415	EMP-LAST-NAME-0415
321	Daniel	Moon
366	Alan	Donovan
371	Beth	Cloud

Department Record

DEPT-ID-0410	5100
DEPT-NAME-0410	Brainstorming
DEPT-HEAD-0410	0015

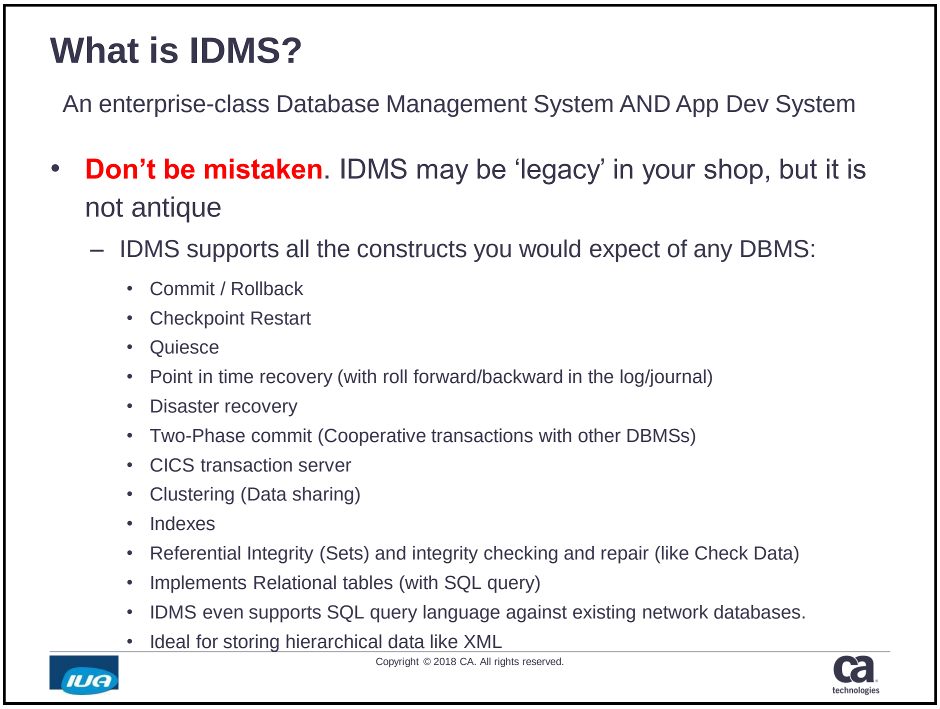
Employee Record(s)

EMP-ID-0415	371
EMP-FIRST-NAME-0415	Beth
EMP-LAST-NAME-0415	Cloud

EMP-ID-0415	366
EMP-FIRST-NAME-0415	Alan
EMP-LAST-NAME-0	Donovan

EMP-ID-0415	321
EMP-FIRST-NAME-0415	Daniel
EMP-LAST-NAME-0415	Moon

End of Set, Area, or Index

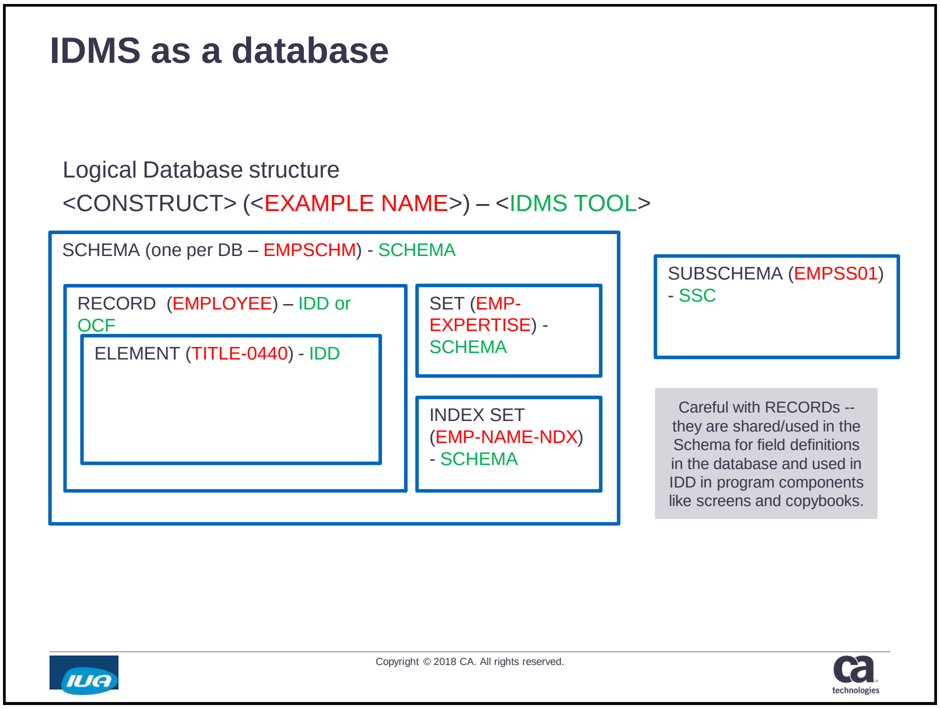


What is IDMS?

An enterprise-class Database Management System AND App Dev System

- **Don’t be mistaken**. IDMS may be ‘legacy’ in your shop, but it is not antique
  - IDMS supports all the constructs you would expect of any DBMS:
    - Commit / Rollback
    - Checkpoint Restart
    - Quiesce
    - Point in time recovery (with roll forward/backward in the log/journal)
    - Disaster recovery
    - Two-Phase commit (Cooperative transactions with other DBMSs)
    - CICS transaction server
    - Clustering (Data sharing)
    - Indexes
    - Referential Integrity (Sets) and integrity checking and repair (like Check Data)
    - Implements Relational tables (with SQL query)
    - IDMS even supports SQL query language against existing network databases.
    - Ideal for storing hierarchical data like XML

IDMS as a Database



IDMS as a database

Logical Database structure

<CONSTRUCT> (<EXAMPLE NAME>) – <IDMS TOOL>

SCHEMA (one per DB – EMPSCHM) - SCHEMA

RECORD (EMPLOYEE) – IDD or OCF

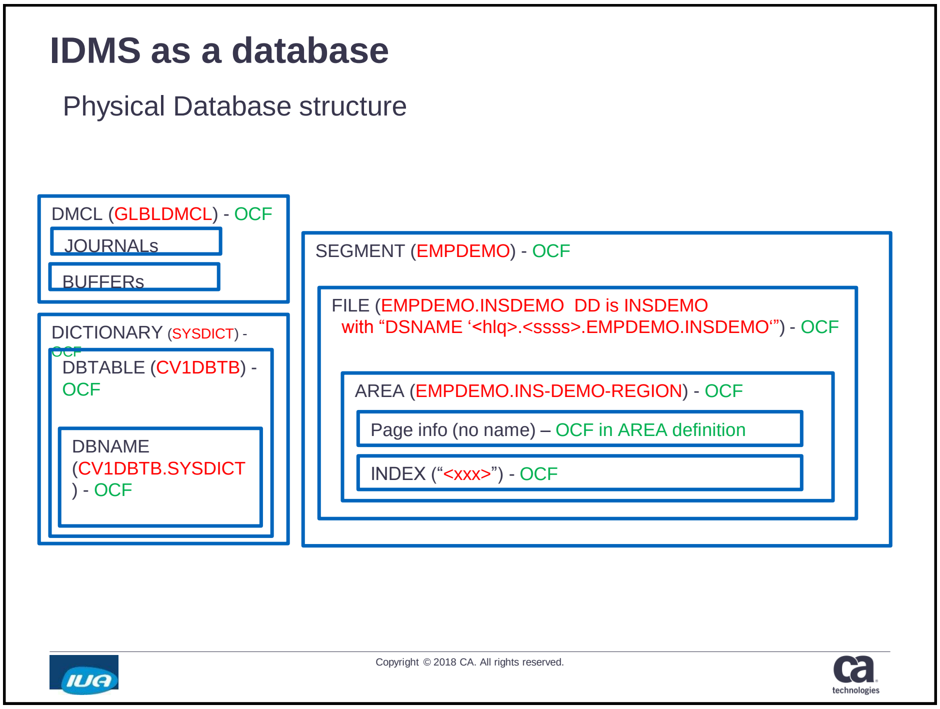
ELEMENT (TITLE-0440) - IDD

SET (EMP-EXPERTISE) - SCHEMA

INDEX SET (EMP-NAME-NDX) - SCHEMA

SUBSCHEMA (EMPSS01) - SSC

Careful with RECORDs -- they are shared/used in the Schema for field definitions in the database and used in IDD in program components like screens and copybooks.



IDMS as a database

Physical Database structure

DMCL (GLBLDMCL) - OCF

JOURNALs

BUFFERs

DICTIONARY (SYSDICT) - OCF

DBTABLE (CV1DBTB) - OCF

DBNAME (CV1DBTB.SYSDICT) - OCF

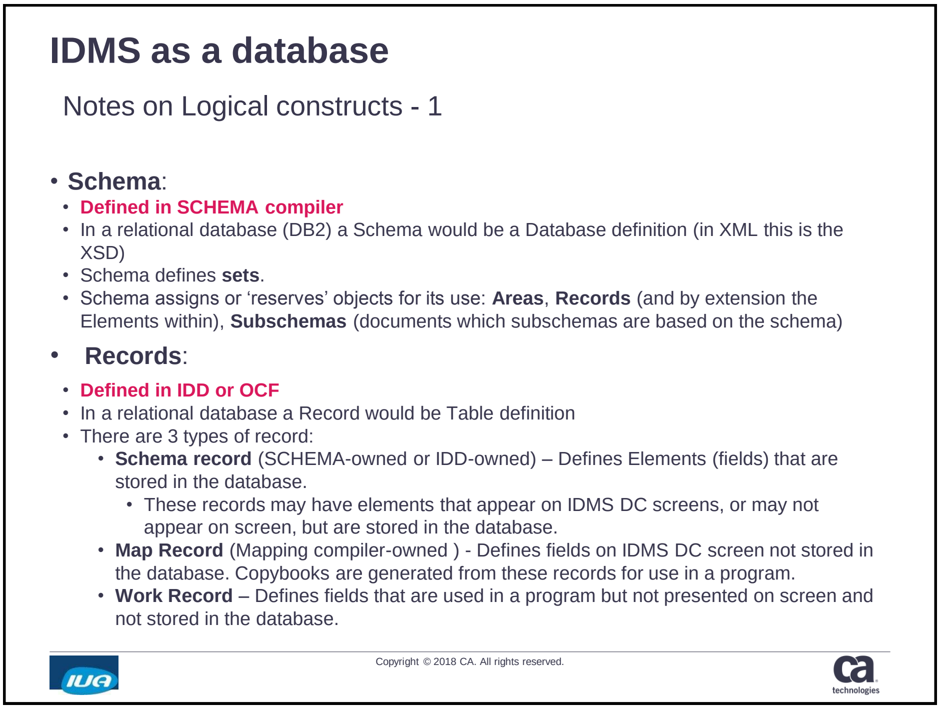
SEGMENT (EMPDEMO) - OCF

FILE (EMPDEMO.INSDEMO DD is INSDEMO with “DSNAME ‘<hlq>.<ssss>.EMPDEMO.INSDEMO’”) - OCF

AREA (EMPDEMO.INS-DEMO-REGION) - OCF

Page info (no name) – OCF in AREA definition

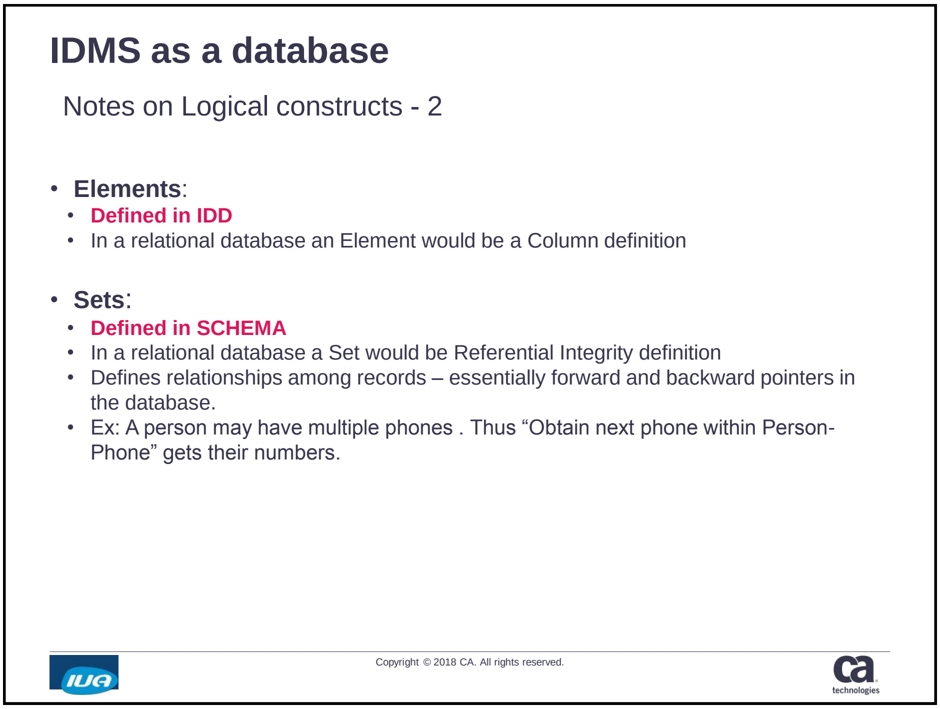
INDEX (“<xxx>”) - OCF



IDMS as a database

Notes on Logical constructs - 1

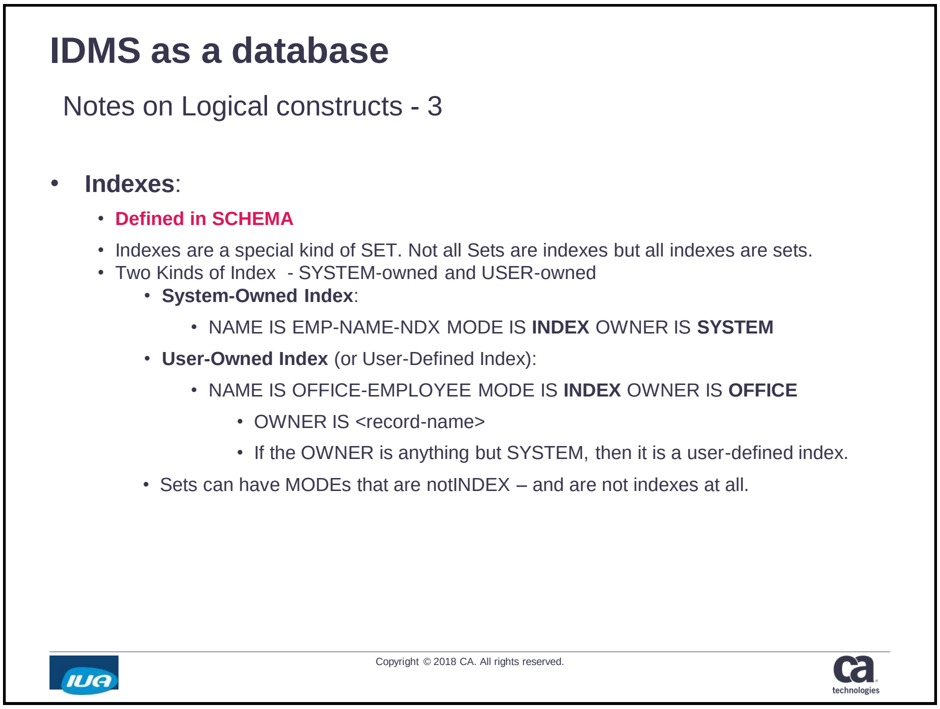
- **Schema**:
  - **Defined in SCHEMA compiler**
  - In a relational database (DB2) a Schema would be a Database definition (in XML this is the XSD)
  - Schema defines **sets**.
  - Schema assigns or ‘reserves’ objects for its use: **Areas**, **Records** (and by extension the Elements within), **Subschemas** (documents which subschemas are based on the schema)
- **Records**:
  - **Defined in IDD or OCF**
  - In a relational database a Record would be Table definition
  - There are 3 types of record:
    - **Schema record** (SCHEMA-owned or IDD-owned) – Defines Elements (fields) that are stored in the database.
      - These records may have elements that appear on IDMS DC screens, or may not appear on screen, but are stored in the database.
    - **Map Record** (Mapping compiler-owned) - Defines fields on IDMS DC screen not stored in the database. Copybooks are generated from these records for use in a program.
    - **Work Record** – Defines fields that are used in a program but not presented on screen and not stored in the database.



IDMS as a database

Notes on Logical constructs - 2

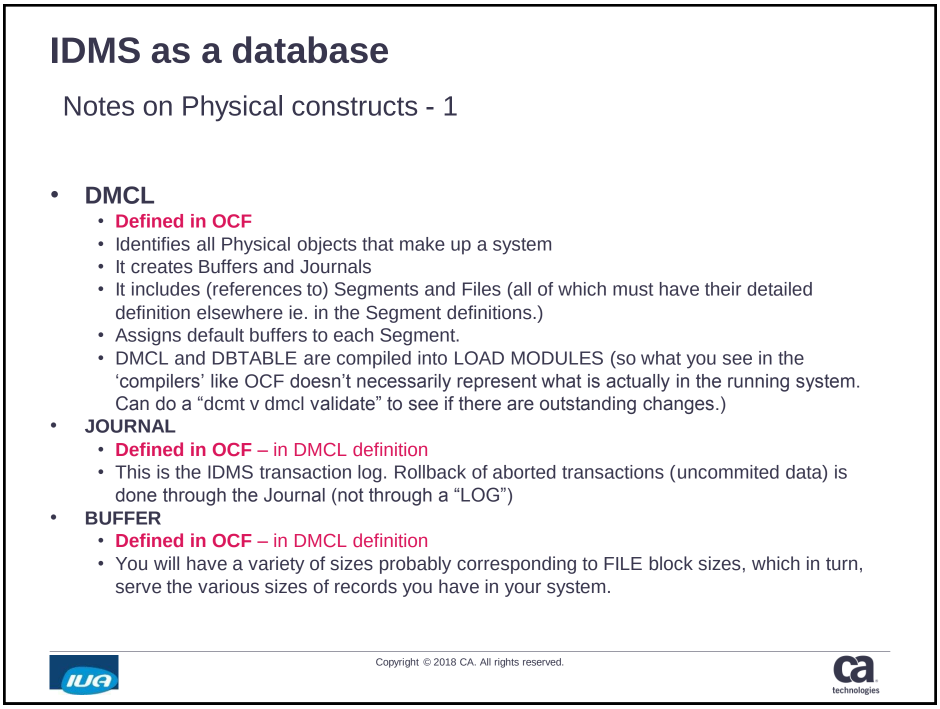
- **Elements**:
  - **Defined in IDD**
  - In a relational database an Element would be a Column definition
- **Sets**:
  - **Defined in SCHEMA**
  - In a relational database a Set would be Referential Integrity definition
  - Defines relationships among records – essentially forward and backward pointers in the database.
  - Ex: A person may have multiple phones . Thus “Obtain next phone within Person-Phone” gets their numbers.



IDMS as a database

Notes on Logical constructs - 3

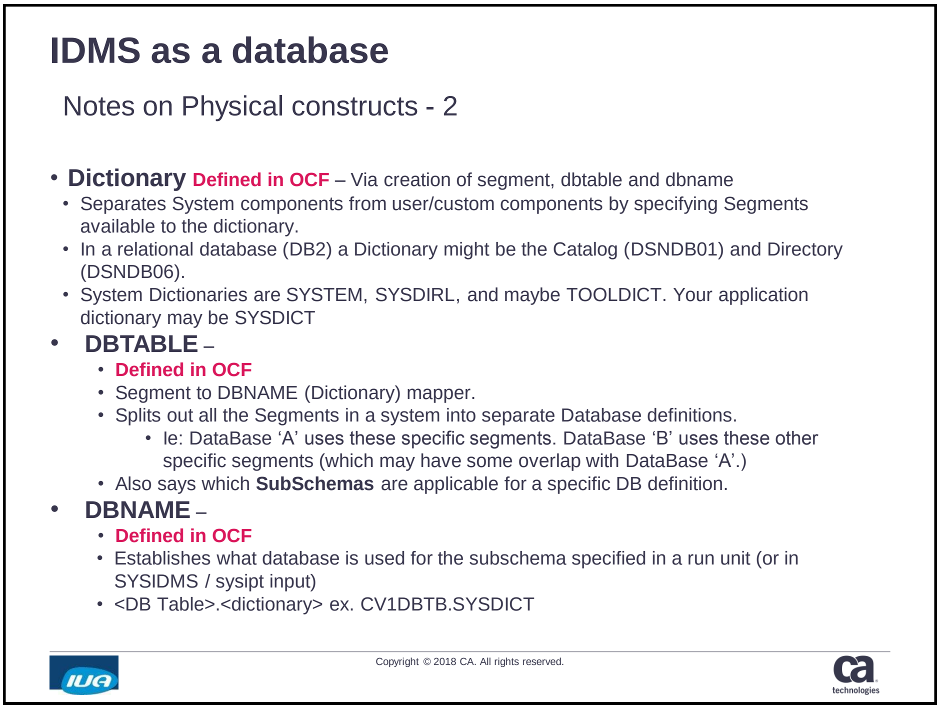
- **Indexes**:
  - **Defined in SCHEMA**
  - Indexes are a special kind of SET. Not all Sets are indexes but all indexes are sets.
  - Two Kinds of Index - SYSTEM-owned and USER-owned
    - **System-Owned Index**:
      - NAME IS EMP-NAME-NDX MODE IS **INDEX** OWNER IS **SYSTEM**
    - **User-Owned Index** (or User-Defined Index):
      - NAME IS OFFICE-EMPLOYEE MODE IS **INDEX** OWNER IS **OFFICE**
        - OWNER IS <record-name>
        - If the OWNER is anything but SYSTEM, then it is a user-defined index.
    - Sets can have MODEs that are notINDEX – and are not indexes at all.



IDMS as a database

Notes on Physical constructs - 1

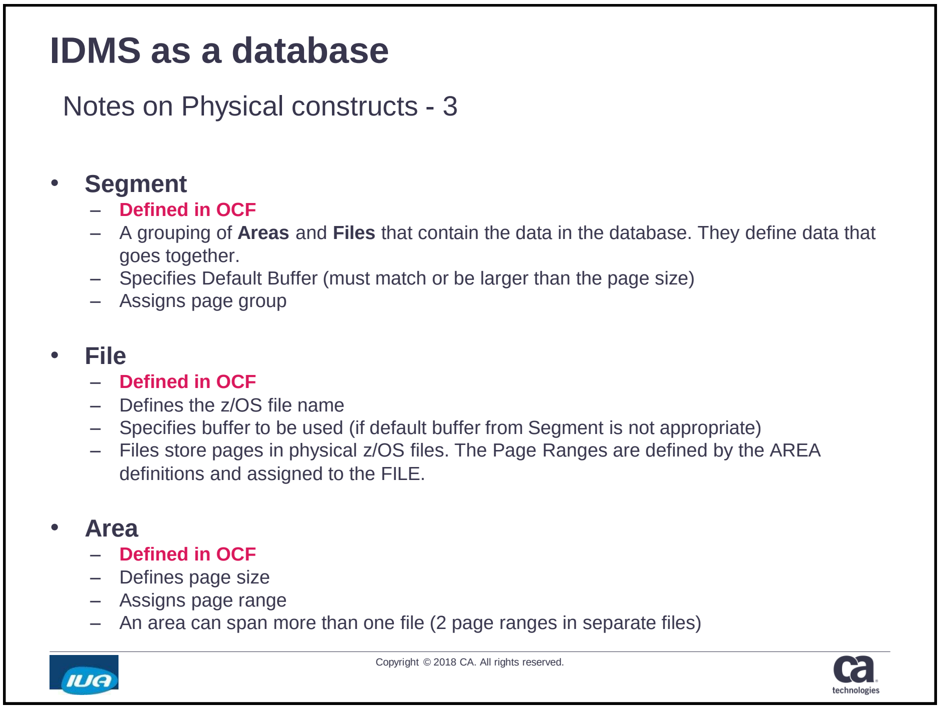
- **DMCL**
  - **Defined in OCF**
  - Identifies all Physical objects that make up a system
  - It creates Buffers and Journals
  - It includes (references to) Segments and Files (all of which must have their detailed definition elsewhere ie. in the Segment definitions.)
  - Assigns default buffers to each Segment.
  - DMCL and DBTABLE are compiled into LOAD MODULES (so what you see in the 'compilers' like OCF doesn't necessarily represent what is actually in the running system. Can do a "dcmt v dmcl validate" to see if there are outstanding changes.)
- **JOURNAL**
  - **Defined in OCF** – in DMCL definition
  - This is the IDMS transaction log. Rollback of aborted transactions (uncommited data) is done through the Journal (not through a "LOG")
- **BUFFER**
  - **Defined in OCF** – in DMCL definition
  - You will have a variety of sizes probably corresponding to FILE block sizes, which in turn, serve the various sizes of records you have in your system.



IDMS as a database

Notes on Physical constructs - 2

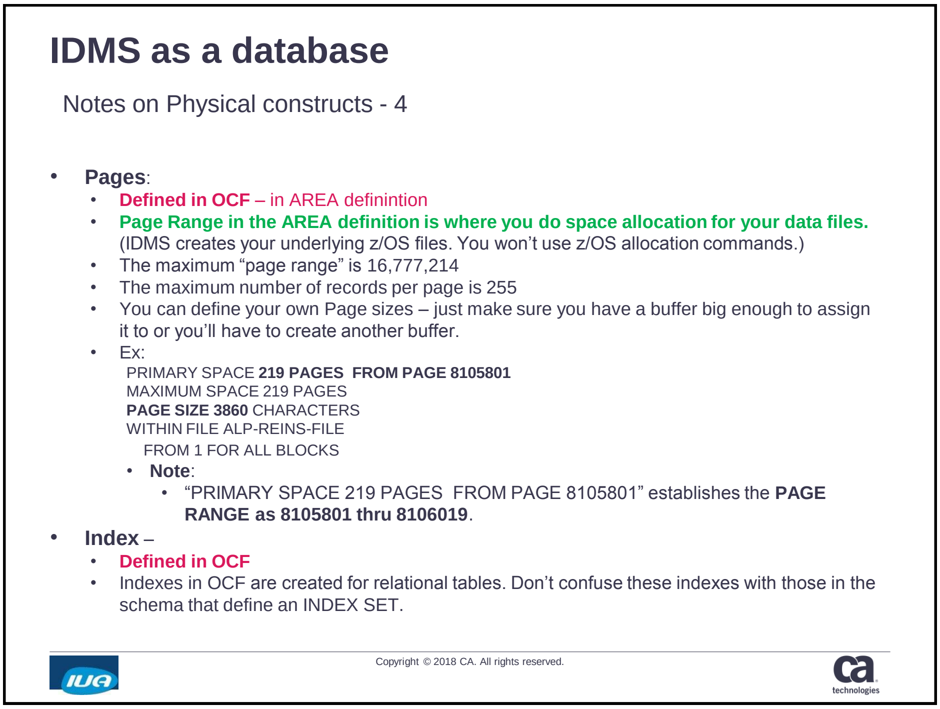
- **Dictionary** **Defined in OCF** – Via creation of segment, dbtable and dbname
  - Separates System components from user/custom components by specifying Segments available to the dictionary.
  - In a relational database (DB2) a Dictionary might be the Catalog (DSNDB01) and Directory (DSNDB06).
  - System Dictionaries are SYSTEM, SYSDIRL, and maybe TOOLDICT. Your application dictionary may be SYSDICT
- **DBTABLE** –
  - **Defined in OCF**
  - Segment to DBNAME (Dictionary) mapper.
  - Splits out all the Segments in a system into separate Database definitions.
    - Ie: DataBase ‘A’ uses these specific segments. DataBase ‘B’ uses these other specific segments (which may have some overlap with DataBase ‘A’.)
  - Also says which **SubSchemas** are applicable for a specific DB definition.
- **DBNAME** –
  - **Defined in OCF**
  - Establishes what database is used for the subschema specified in a run unit (or in SYSIDMS / sysipt input)
  - <DB Table>.<dictionary> ex. CV1DBTB.SYSDICT



IDMS as a database

Notes on Physical constructs - 3

- **Segment**
  - **Defined in OCF**
  - A grouping of **Areas** and **Files** that contain the data in the database. They define data that goes together.
  - Specifies Default Buffer (must match or be larger than the page size)
  - Assigns page group
- **File**
  - **Defined in OCF**
  - Defines the z/OS file name
  - Specifies buffer to be used (if default buffer from Segment is not appropriate)
  - Files store pages in physical z/OS files. The Page Ranges are defined by the AREA definitions and assigned to the FILE.
- **Area**
  - **Defined in OCF**
  - Defines page size
  - Assigns page range
  - An area can span more than one file (2 page ranges in separate files)



IDMS as a database

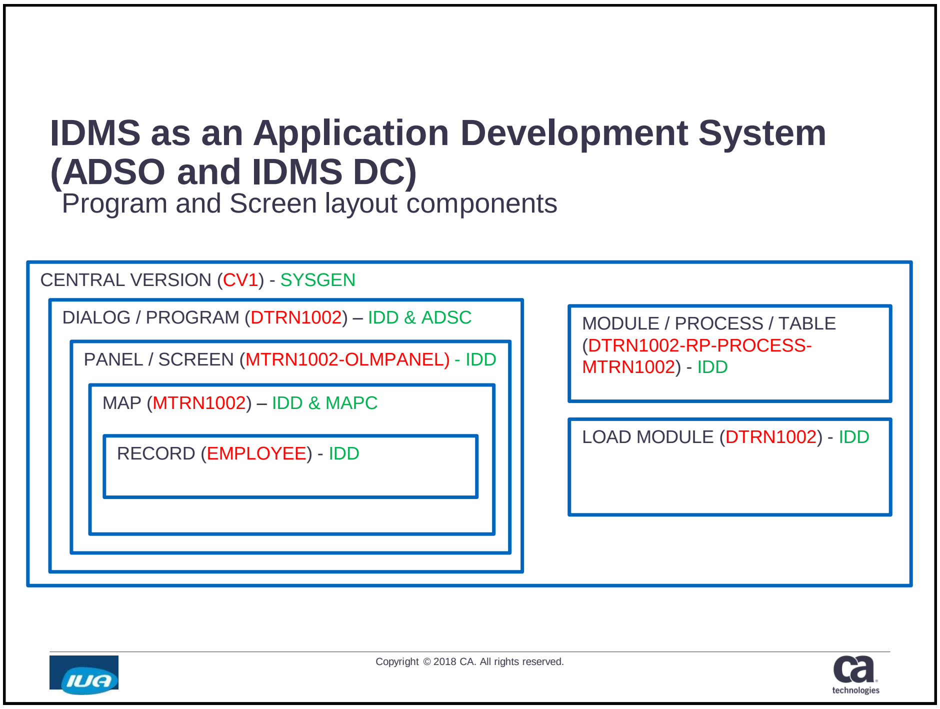
Notes on Physical constructs - 4

- **Pages**:
  - **Defined in OCF** – in AREA definintion
  - **Page Range in the AREA definition is where you do space allocation for your data files.** (IDMS creates your underlying z/OS files. You won't use z/OS allocation commands.)
  - The maximum "page range" is 16,777,214
  - The maximum number of records per page is 255
  - You can define your own Page sizes – just make sure you have a buffer big enough to assign it to or you'll have to create another buffer.
  - Ex:

    PRIMARY SPACE **219 PAGES FROM PAGE 8105801**
    MAXIMUM SPACE 219 PAGES
    PAGE SIZE 3860 CHARACTERS
    WITHIN FILE ALP-REINS-FILE
    FROM 1 FOR ALL BLOCKS

    - **Note**:
      - "PRIMARY SPACE 219 PAGES FROM PAGE 8105801" establishes the **PAGE RANGE as 8105801 thru 8106019**.
- **Index** –
  - **Defined in OCF**
  - Indexes in OCF are created for relational tables. Don't confuse these indexes with those in the schema that define an INDEX SET.

IDMS as an Application Development System



IDMS as an Application Development System (ADSO and IDMS DC)

Program and Screen layout components

CENTRAL VERSION (CV1) - SYSGEN

DIALOG / PROGRAM (DTRN1002) – IDD & ADSC

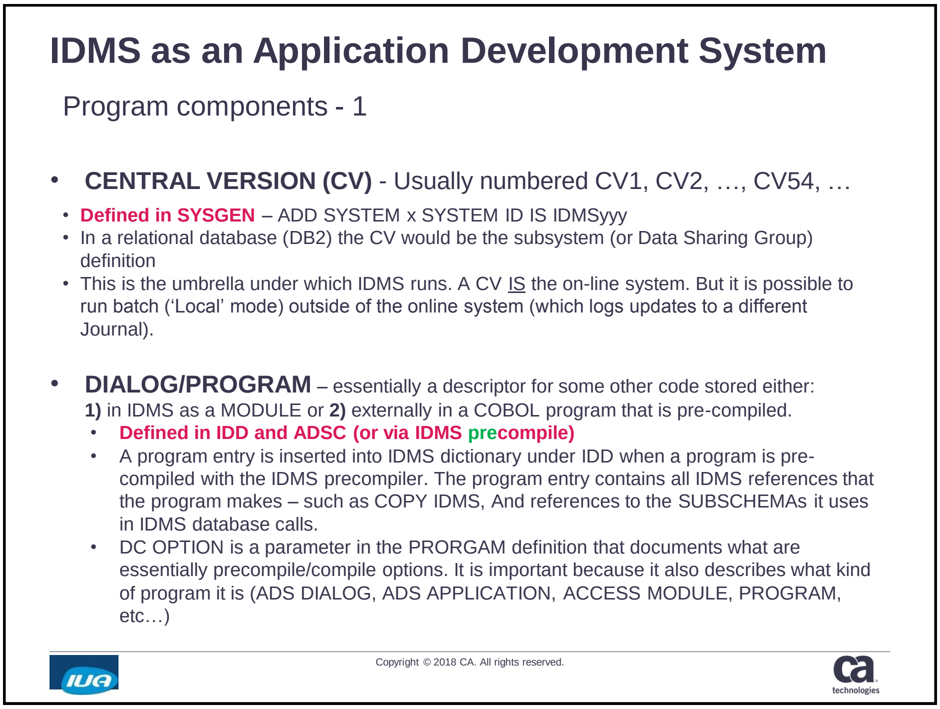
PANEL / SCREEN (MTRN1002-OLMPANEL) - IDD

MAP (MTRN1002) – IDD & MAPC

RECORD (EMPLOYEE) - IDD

MODULE / PROCESS / TABLE (DTRN1002-RP-PROCESS-MTRN1002) - IDD

LOAD MODULE (DTRN1002) - IDD

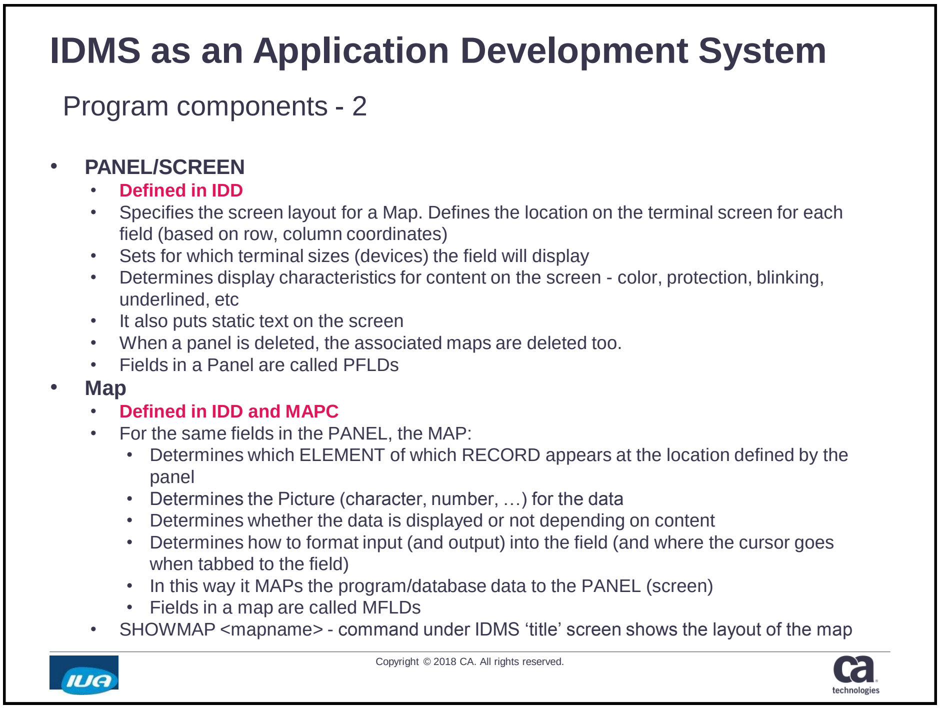


IDMS as an Application Development System

Program components - 1

- **CENTRAL VERSION (CV)** - Usually numbered CV1, CV2, …, CV54, …
  - **Defined in SYSGEN** – ADD SYSTEM x SYSTEM ID IS IDMSyyy
  - In a relational database (DB2) the CV would be the subsystem (or Data Sharing Group) definition
  - This is the umbrella under which IDMS runs. A CV IS the on-line system. But it is possible to run batch ('Local' mode) outside of the online system (which logs updates to a different Journal).

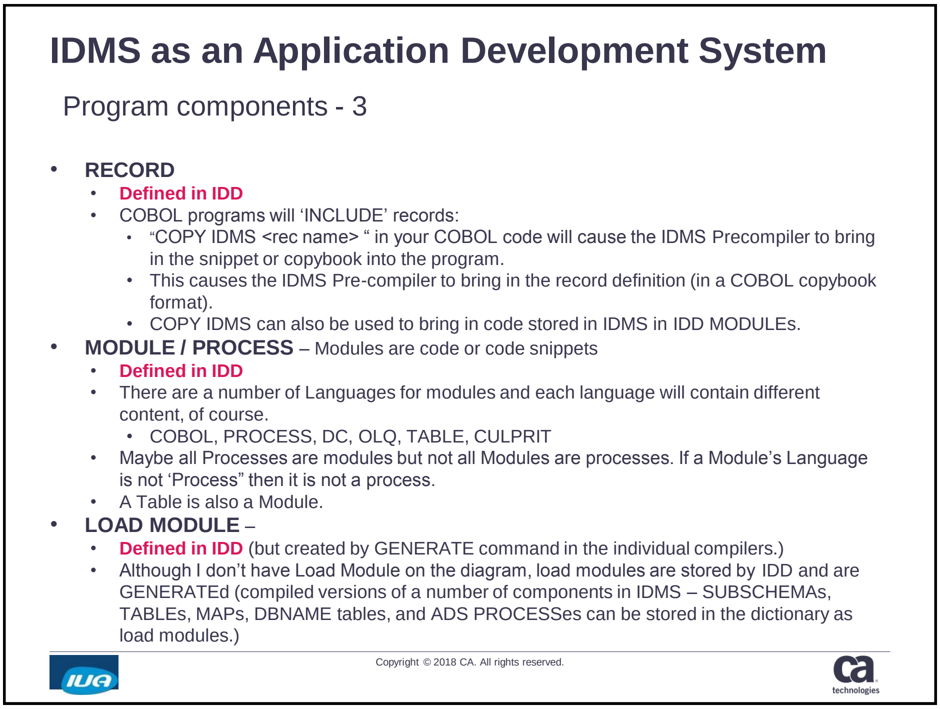
- **DIALOG/PROGRAM** – essentially a descriptor for some other code stored either: **1)** in IDMS as a MODULE or **2)** externally in a COBOL program that is pre-compiled.
  - **Defined in IDD and ADSC (or via IDMS precompile)**
  - A program entry is inserted into IDMS dictionary under IDD when a program is pre-compiled with the IDMS precompiler. The program entry contains all IDMS references that the program makes – such as COPY IDMS, And references to the SUBSCHEMAs it uses in IDMS database calls.
  - DC OPTION is a parameter in the PRORGAM definition that documents what are essentially precompile/compile options. It is important because it also describes what kind of program it is (ADS DIALOG, ADS APPLICATION, ACCESS MODULE, PROGRAM, etc…)



IDMS as an Application Development System

Program components - 2

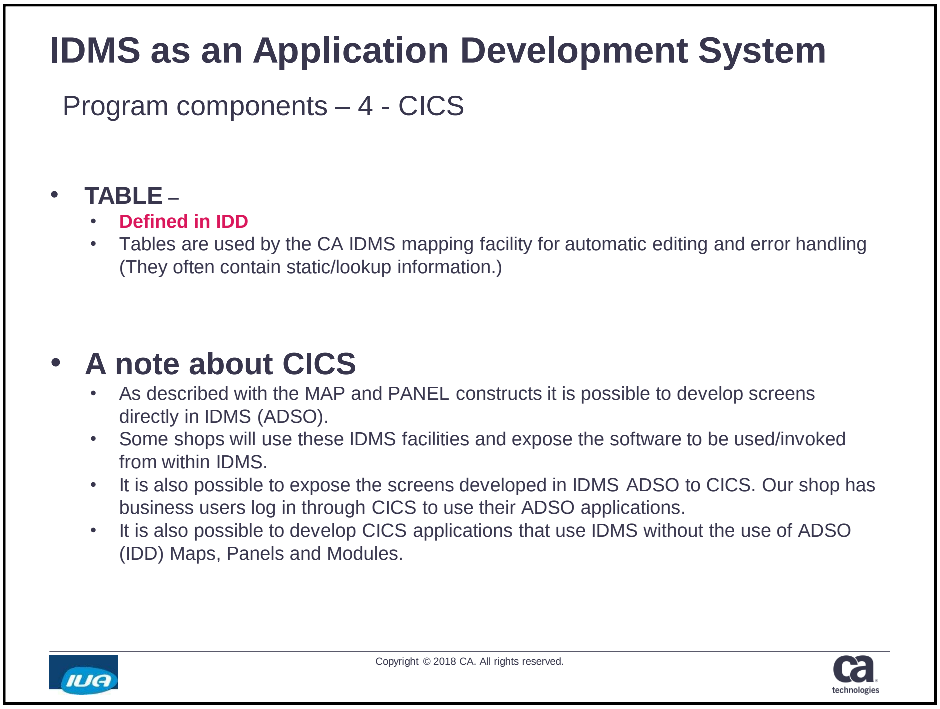
- **PANEL/SCREEN**
  - **Defined in IDD**
  - Specifies the screen layout for a Map. Defines the location on the terminal screen for each field (based on row, column coordinates)
  - Sets for which terminal sizes (devices) the field will display
  - Determines display characteristics for content on the screen - color, protection, blinking, underlined, etc
  - It also puts static text on the screen
  - When a panel is deleted, the associated maps are deleted too.
  - Fields in a Panel are called PFLDs
- **Map**
  - **Defined in IDD and MAPC**
  - For the same fields in the PANEL, the MAP:
    - Determines which ELEMENT of which RECORD appears at the location defined by the panel
    - Determines the Picture (character, number, …) for the data
    - Determines whether the data is displayed or not depending on content
    - Determines how to format input (and output) into the field (and where the cursor goes when tabbed to the field)
    - In this way it MAPs the program/database data to the PANEL (screen)
    - Fields in a map are called MFLDs
  - SHOWMAP <mapname> - command under IDMS ‘title’ screen shows the layout of the map



IDMS as an Application Development System

Program components - 3

- **RECORD**
  - **Defined in IDD**
  - COBOL programs will ‘INCLUDE’ records:
    - “COPY IDMS <rec name> “ in your COBOL code will cause the IDMS Precompiler to bring in the snippet or copybook into the program.
    - This causes the IDMS Pre-compiler to bring in the record definition (in a COBOL copybook format).
    - COPY IDMS can also be used to bring in code stored in IDMS in IDD MODULEs.
- **MODULE / PROCESS** – Modules are code or code snippets
  - **Defined in IDD**
  - There are a number of Languages for modules and each language will contain different content, of course.
    - COBOL, PROCESS, DC, OLQ, TABLE, CULPRIT
  - Maybe all Processes are modules but not all Modules are processes. If a Module’s Language is not ‘Process” then it is not a process.
  - A Table is also a Module.
- **LOAD MODULE** –
  - **Defined in IDD** (but created by GENERATE command in the individual compilers.)
  - Although I don’t have Load Module on the diagram, load modules are stored by IDD and are GENERATEd (compiled versions of a number of components in IDMS – SUBSCHEMAs, TABLEs, MAPs, DBNAME tables, and ADS PROCESSes can be stored in the dictionary as load modules.)

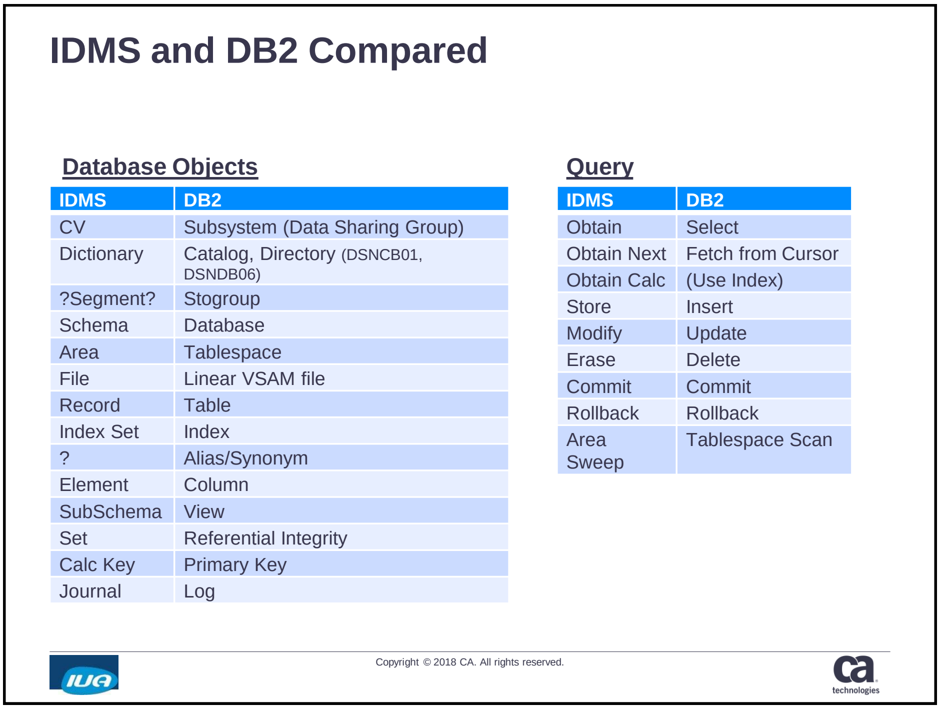


IDMS as an Application Development System

Program components – 4 - CICS

- **TABLE** –
  - **Defined in IDD**
  - Tables are used by the CA IDMS mapping facility for automatic editing and error handling (They often contain static/lookup information.)

- **A note about CICS**
  - As described with the MAP and PANEL constructs it is possible to develop screens directly in IDMS (ADSO).
  - Some shops will use these IDMS facilities and expose the software to be used/invoked from within IDMS.
  - It is also possible to expose the screens developed in IDMS ADSO to CICS. Our shop has business users log in through CICS to use their ADSO applications.
  - It is also possible to develop CICS applications that use IDMS without the use of ADSO (IDD) Maps, Panels and Modules.



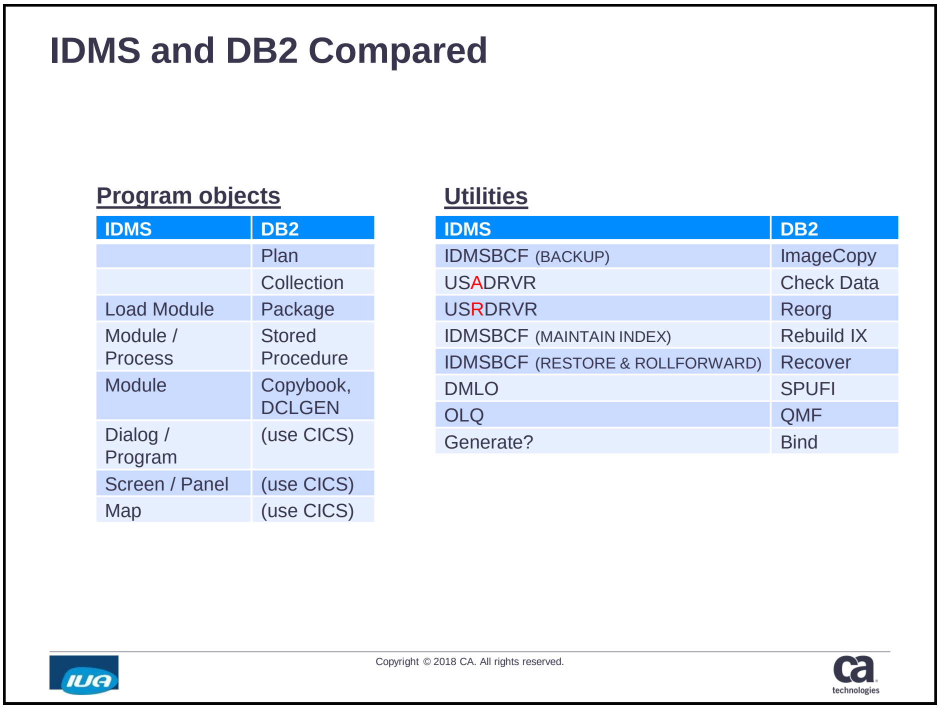
IDMS and DB2 Compared

Database Objects

IDMS	DB2
CV	Subsystem (Data Sharing Group)
Dictionary	Catalog, Directory (DSNCB01, DSNDB06)
?Segment?	Stogroup
Schema	Database
Area	Tablespace
File	Linear VSAM file
Record	Table
Index Set	Index
?	Alias/Synonym
Element	Column
SubSchema	View
Set	Referential Integrity
Calc Key	Primary Key
Journal	Log

Query

IDMS	DB2
Obtain	Select
Obtain Next	Fetch from Cursor
Obtain Calc	(Use Index)
Store	Insert
Modify	Update
Erase	Delete
Commit	Commit
Rollback	Rollback
Area Sweep	Tablespace Scan



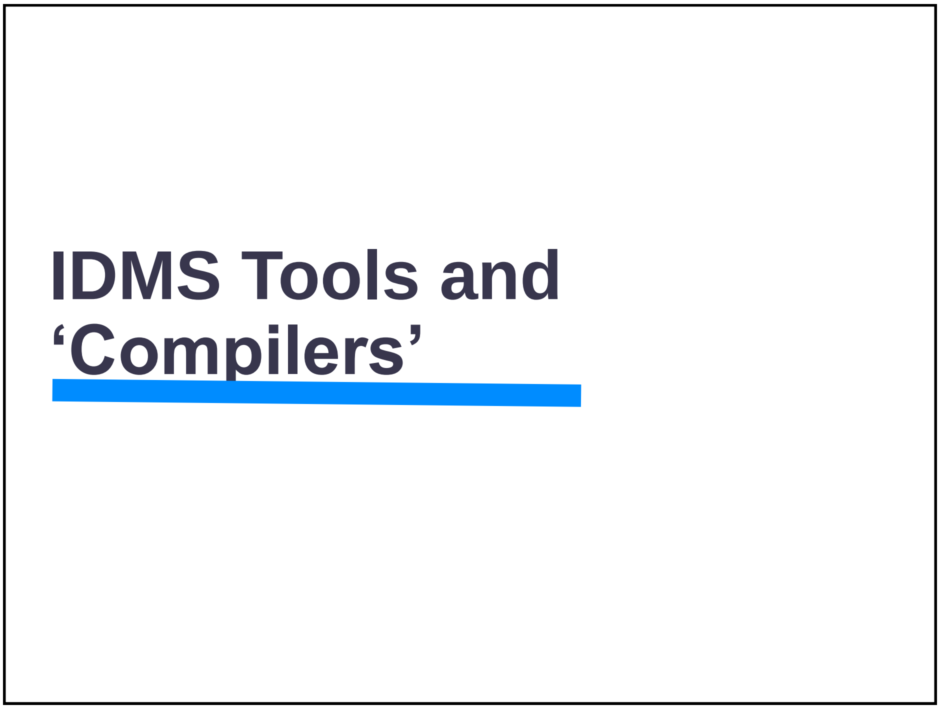
IDMS and DB2 Compared

Program objects

IDMS	DB2
	Plan
	Collection
Load Module	Package
Module / Process	Stored Procedure
Module	Copybook, DCLGEN
Dialog / Program	(use CICS)
Screen / Panel	(use CICS)
Map	(use CICS)

Utilities

IDMS	DB2
IDMSBCF (BACKUP)	ImageCopy
USADRVR	Check Data
USRDRVR	Reorg
IDMSBCF (MAINTAIN INDEX)	Rebuild IX
IDMSBCF (RESTORE & ROLLFORWARD)	Recover
DMLO	SPUFI
OLQ	QMF
Generate?	Bind



IDMS Tools and ‘Compilers’

Interacting with IDMS as a DBA

```
 V1   ENTER NEXT TASK CODE:       CA IDMS release 18.5 tape GJI50B node IDMSDCT

                         HELP  !!!

 Te                                                      R   2 C   2  TCPI0472
```

IDMS Tools and 'Compilers'

The "Compilers" – the ones you'll use the most – and what they are for

- My mentor called them 'Compilers'. I believe CA calls them '**online CA IDMS components**'.

PROGRAM Component compilers
- IDD (or IDDM – menu)
- MAPC
- ADSA
- ADSC
- ADS
- ADSM

Logical DB Component compilers
- SCHEMA
- SubSchema Compiler (SSC)

Physical DB component compilers
- OCF

Using the Tools

The “Compilers” - Common controls among all (most) compilers

Get to know your PA-1 key and Clear key!!

Mine are Alt-Insert and Alt-z respectively

- Display (and Display as syntax)
- Punch (Useful in batch to ‘export’ definitions into a file)
- SUSpend
- Find command
- TCF (Transfer control facility. A step above SUSpend)
- SIGNON USAGE RET (Start a read only session)

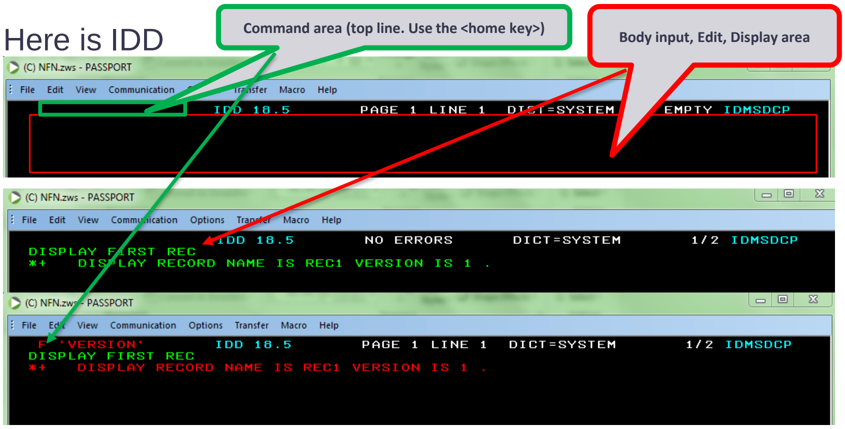
How do I get out !! ??

BYE, SIGNOFF, LOGOFF, EXIT, QUIT, END, Clear-key, PA-1

Using the Tools

The “Compilers” – Example: Top line commands vs second line / body commands

Here is IDD


Command area (top line. Use the <home key>)

Body input, Edit, Display area

(C) NFN.zws - PASSPORT
File Edit View Communication Transfer Macro Help

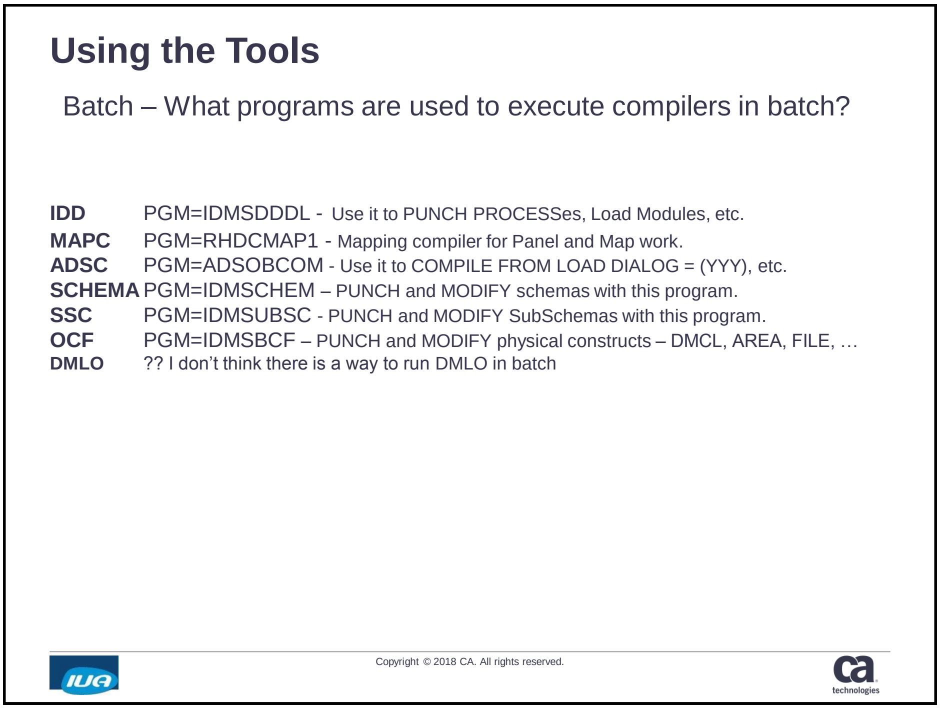
```
               IDD 18.5          PAGE 1 LINE 1   DICT=SYSTEM      EMPTY IDMSDCP
```

(C) NFN.zws - PASSPORT
File Edit View Communication Options Transfer Macro Help

```
                IDD 18.5          NO ERRORS        DICT=SYSTEM          1/2 IDMSDCP
DISPLAY FIRST REC
*+   DISPLAY RECORD NAME IS REC1 VERSION IS 1 .
```

(C) NFN.zws - PASSPORT
File Edit View Communication Options Transfer Macro Help

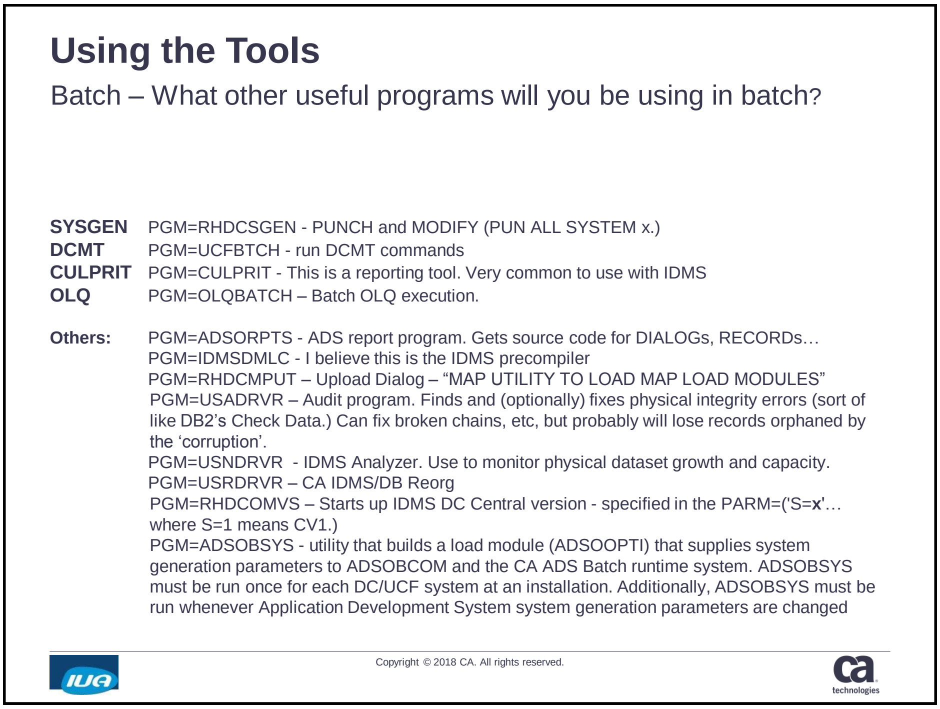
```
 F 'VERSION'      IDD 18.5          PAGE 1 LINE 1   DICT=SYSTEM          1/2 IDMSDCP
DISPLAY FIRST REC
*+   DISPLAY RECORD NAME IS REC1 VERSION IS 1 .
```



Using the Tools

Batch – What programs are used to execute compilers in batch?

IDD PGM=IDMSDDDL - Use it to PUNCH PROCESSes, Load Modules, etc.
MAPC PGM=RHDCMAP1 - Mapping compiler for Panel and Map work.
ADSC PGM=ADSOBCOM - Use it to COMPILE FROM LOAD DIALOG = (YYY), etc.
SCHEMA PGM=IDMSCHEM – PUNCH and MODIFY schemas with this program.
SSC PGM=IDMSUBSC - PUNCH and MODIFY SubSchemas with this program.
OCF PGM=IDMSBCF – PUNCH and MODIFY physical constructs – DMCL, AREA, FILE, …
DMLO ?? I don’t think there is a way to run DMLO in batch

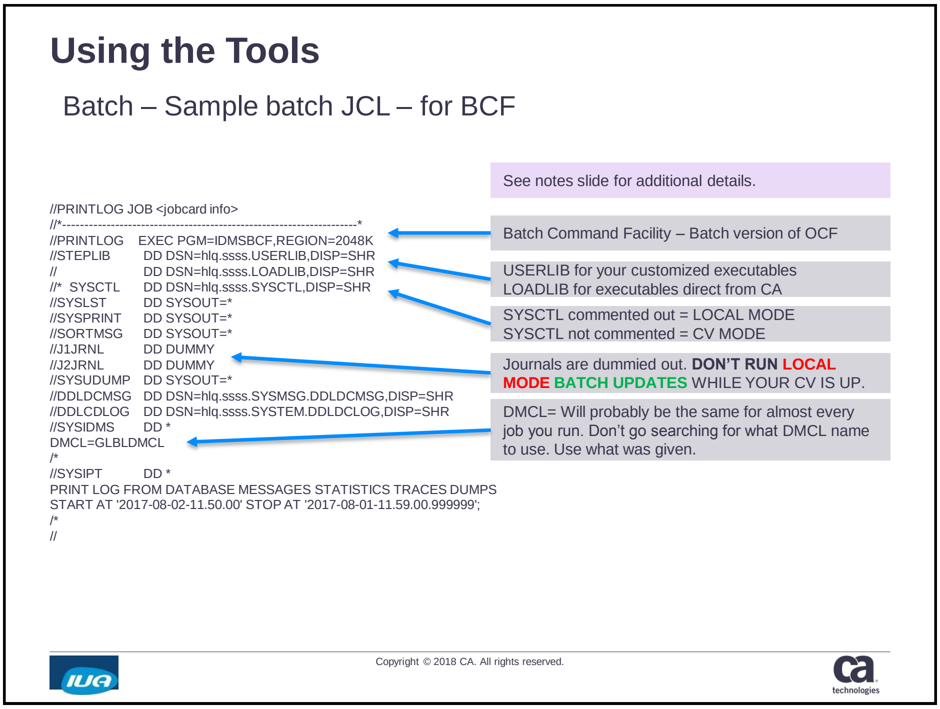


Using the Tools

Batch – What other useful programs will you be using in batch?

SYSGEN PGM=RHDCSGEN - PUNCH and MODIFY (PUN ALL SYSTEM x.)
DCMT PGM=UCFBTCH - run DCMT commands
CULPRIT PGM=CULPRIT - This is a reporting tool. Very common to use with IDMS
OLQ PGM=OLQBATCH – Batch OLQ execution.

Others: PGM=ADSORPTS - ADS report program. Gets source code for DIALOGs, RECORDs…
PGM=IDMSDMLC - I believe this is the IDMS precompiler
PGM=RHDCMPUT – Upload Dialog – “MAP UTILITY TO LOAD MAP LOAD MODULES”
PGM=USADRVR – Audit program. Finds and (optionally) fixes physical integrity errors (sort of like DB2’s Check Data.) Can fix broken chains, etc, but probably will lose records orphaned by the ‘corruption’.
PGM=USNDRVR - IDMS Analyzer. Use to monitor physical dataset growth and capacity.
PGM=USRDRVR – CA IDMS/DB Reorg
PGM=RHDCOMVS – Starts up IDMS DC Central version - specified in the PARM=('S=**x**'… where S=1 means CV1.)
PGM=ADSOBSYS - utility that builds a load module (ADSOOPTI) that supplies system generation parameters to ADSOBCOM and the CA ADS Batch runtime system. ADSOBSYS must be run once for each DC/UCF system at an installation. Additionally, ADSOBSYS must be run whenever Application Development System system generation parameters are changed



Using the Tools

Batch – Sample batch JCL – for BCF

See notes slide for additional details.

```
//PRINTLOG JOB <jobcard info>
//*-----------------------------------------------------------------*
//PRINTLOG    EXEC PGM=IDMSBCF,REGION=2048K
//STEPLIB       DD DSN=hlq.ssss.USERLIB,DISP=SHR
//              DD DSN=hlq.ssss.LOADLIB,DISP=SHR
//*  SYSCTL     DD DSN=hlq.ssss.SYSCTL,DISP=SHR
//SYSLST        DD SYSOUT=*
//SYSPRINT      DD SYSOUT=*
//SORTMSG       DD SYSOUT=*
//J1JRNL        DD DUMMY
//J2JRNL        DD DUMMY
//SYSUDUMP      DD SYSOUT=*
//DDLDCMSG      DD DSN=hlq.ssss.SYSMSG.DDLDCMSG,DISP=SHR
//DDLCDLOG      DD DSN=hlq.ssss.SYSTEM.DDLDCLOG,DISP=SHR
//SYSIDMS       DD *
DMCL=GLBLDMCL
/*
//SYSIPT        DD *
PRINT LOG FROM DATABASE MESSAGES STATISTICS TRACES DUMPS
START AT '2017-08-02-11.50.00' STOP AT '2017-08-01-11.59.00.999999';
/*
//
```

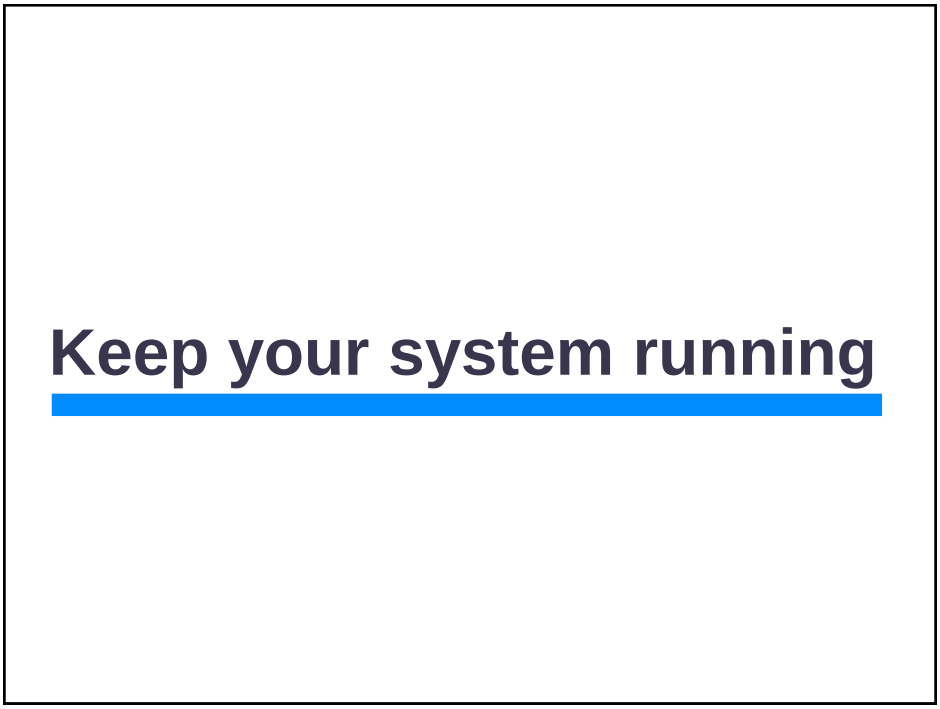
Batch Command Facility – Batch version of OCF

USERLIB for your customized executables
LOADLIB for executables direct from CA

SYSCTL commented out = LOCAL MODE
SYSCTL not commented = CV MODE

Journals are dummied out. **DON'T RUN LOCAL MODE BATCH UPDATES** WHILE YOUR CV IS UP.

DMCL= Will probably be the same for almost every job you run. Don't go searching for what DMCL name to use. Use what was given.



Keep your system running

Keeping your system running

Basics – More 'Compilers'

Full Screen commands / utils

- SYSGEN
- OLP
- DMLO (Ad-hoc query – like SPUFI in the DB2 world)
- Oper Watch
- Lockmon
- OLQ – more of a reporting language

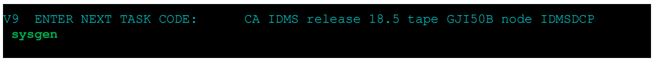
ENTER NEXT TASK CODE - Commands

- DCPROFILE
- DCMT
- Look
- DCUF
- …

Keeping your system running

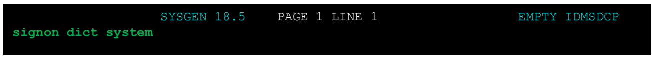
SYSGEN example

1



```
V9  ENTER NEXT TASK CODE:      CA IDMS release 18.5 tape GJI50B node IDMSDCP
 sysgen
```

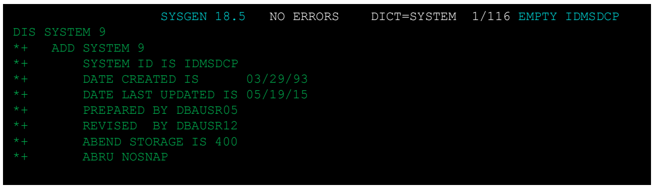
2



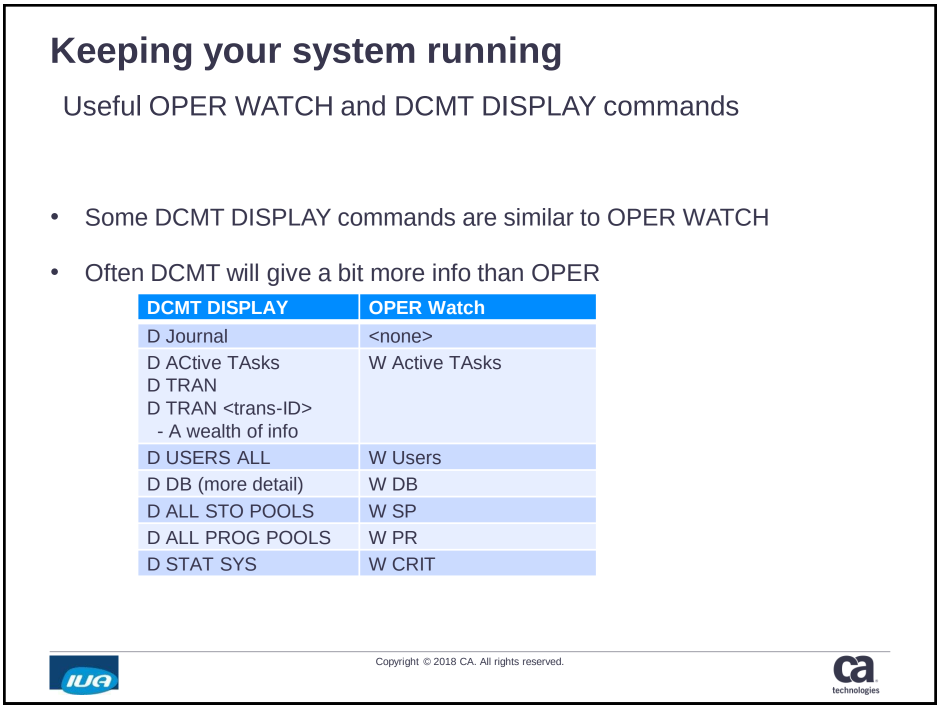
```
                    SYSGEN 18.5    PAGE 1 LINE 1                 EMPTY IDMSDCP
 signon dict system
```

DISPLAY … AS SYN
-- Then you can use line commands to edit the syntax and 'submit' it

%C - copy a line
%B or **%A** - put the copied line before or after this line.
%R - repeat a line
%D - delete a line
%M - Move a line
%t - moves this line to top of screen



```
                    SYSGEN 18.5   NO ERRORS    DICT=SYSTEM  1/116 EMPTY IDMSDCP
 DIS SYSTEM 9
 *+   ADD SYSTEM 9
 *+       SYSTEM ID IS IDMSDCP
 *+       DATE CREATED IS      03/29/93
 *+       DATE LAST UPDATED IS 05/19/15
 *+       PREPARED BY DBAUSR05
 *+       REVISED  BY DBAUSR12
 *+       ABEND STORAGE IS 400
 *+       ABRU NOSNAP
```

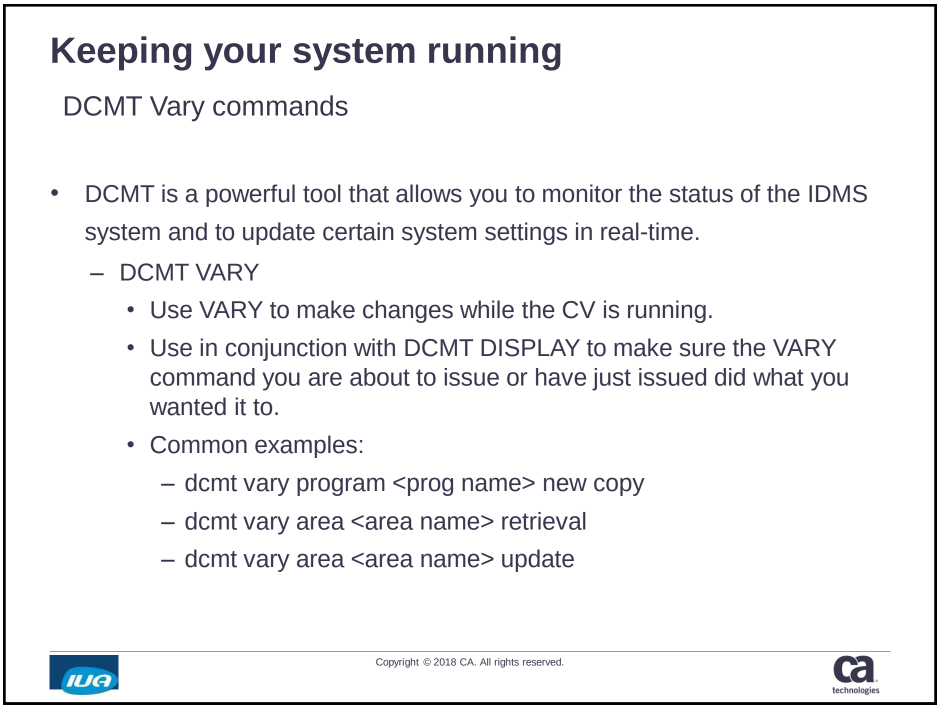


Keeping your system running

Useful OPER WATCH and DCMT DISPLAY commands

- Some DCMT DISPLAY commands are similar to OPER WATCH
- Often DCMT will give a bit more info than OPER

DCMT DISPLAY	OPER Watch
D Journal	<none>
D ACtive TAsks D TRAN D TRAN <trans-ID> - A wealth of info	W Active TAsks
D USERS ALL	W Users
D DB (more detail)	W DB
D ALL STO POOLS	W SP
D ALL PROG POOLS	W PR
D STAT SYS	W CRIT



Keeping your system running

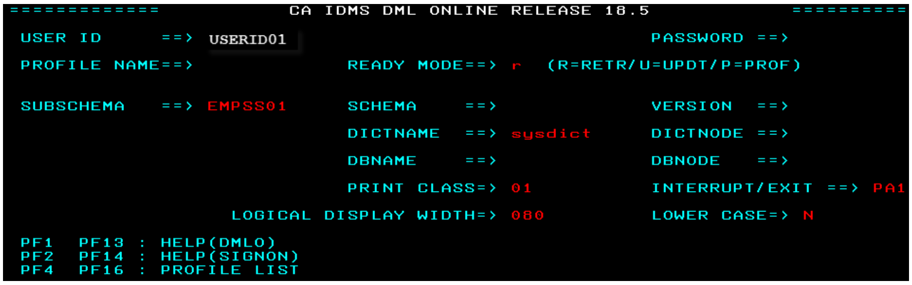
DCMT Vary commands

- DCMT is a powerful tool that allows you to monitor the status of the IDMS system and to update certain system settings in real-time.
  - DCMT VARY
    - Use VARY to make changes while the CV is running.
    - Use in conjunction with DCMT DISPLAY to make sure the VARY command you are about to issue or have just issued did what you wanted it to.
    - Common examples:
      - dcmt vary program <prog name> new copy
      - dcmt vary area <area name> retrieval
      - dcmt vary area <area name> update

Keeping your system running

DMLO ‘query’ example – Using the IDMS Demo Employee DB

DMLO Entry Screen:

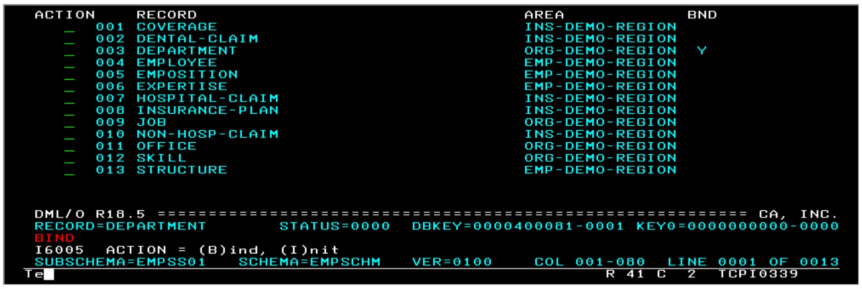


```
=============              CA IDMS DML ONLINE RELEASE 18.5              ==========

 USER ID      ==> USERID01                           PASSWORD ==>
 PROFILE NAME==>              READY MODE==> r  (R=RETR/U=UPDT/P=PROF)

 SUBSCHEMA    ==> EMPSS01      SCHEMA     ==>            VERSION  ==>
                               DICTNAME   ==> sysdict    DICTNODE ==>
                               DBNAME     ==>            DBNODE   ==>
                               PRINT CLASS=> 01          INTERRUPT/EXIT ==> PA1
                    LOGICAL DISPLAY WIDTH=> 080          LOWER CASE=> N

 PF1  PF13 : HELP(DMLO)
 PF2  PF14 : HELP(SIGNON)
 PF4  PF16 : PROFILE LIST
```

Keeping your system running

DMLO ‘query’ example

DMLO - BIND

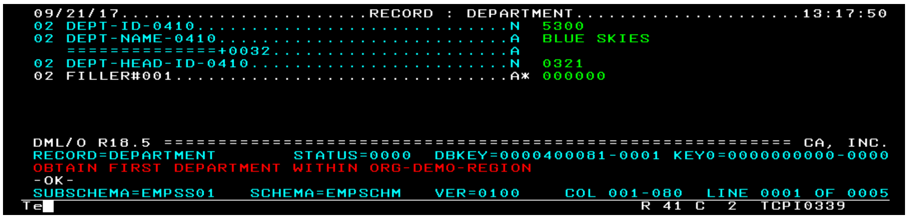


```
  ACTION     RECORD                                          AREA            BND
     _    001 COVERAGE                                       INS-DEMO-REGION
     _    002 DENTAL-CLAIM                                   INS-DEMO-REGION
     _    003 DEPARTMENT                                     ORG-DEMO-REGION  Y
     _    004 EMPLOYEE                                       EMP-DEMO-REGION
     _    005 EMPOSITION                                     EMP-DEMO-REGION
     _    006 EXPERTISE                                      EMP-DEMO-REGION
     _    007 HOSPITAL-CLAIM                                 INS-DEMO-REGION
     _    008 INSURANCE-PLAN                                 INS-DEMO-REGION
     _    009 JOB                                            ORG-DEMO-REGION
     _    010 NON-HOSP-CLAIM                                 INS-DEMO-REGION
     _    011 OFFICE                                         ORG-DEMO-REGION
     _    012 SKILL                                          ORG-DEMO-REGION
     _    013 STRUCTURE                                      EMP-DEMO-REGION


  DML/O R18.5 ========================================================= CA, INC.
  RECORD=DEPARTMENT          STATUS=0000  DBKEY=0000400081-0001 KEY0=0000000000-0000
  BIND
  I6005  ACTION = (B)ind, (I)nit
  SUBSCHEMA=EMPSS01   SCHEMA=EMPSCHM   VER=0100    COL 001-080  LINE 0001 OF 0013
 Te                                                    R 41 C  2  TCPI0339
```

Keeping your system running

DMLO ‘query’ example

DMLO - Obtain



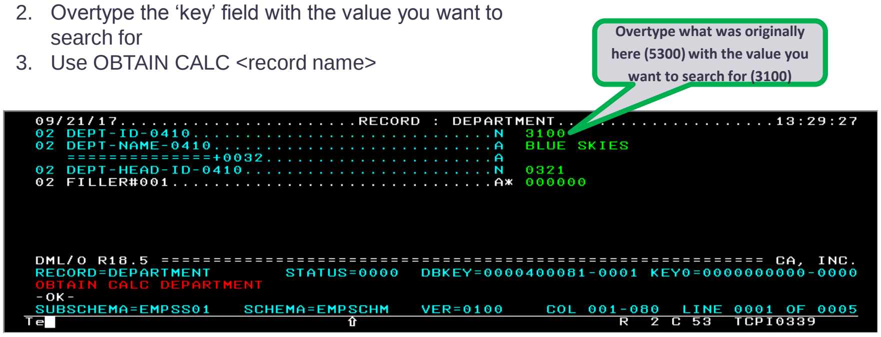
```
 09/21/17.......................RECORD : DEPARTMENT......................13:17:50
 02 DEPT-ID-0410..............................N  5300
 02 DEPT-NAME-0410............................A  BLUE SKIES
    ==============+0032.......................A
 02 DEPT-HEAD-ID-0410.........................N  0321
 02 FILLER#001................................A* 000000



 DML/O R18.5 =========================================================== CA, INC.
 RECORD=DEPARTMENT       STATUS=0000   DBKEY=0000400081-0001 KEY0=0000000000-0000
 OBTAIN FIRST DEPARTMENT WITHIN ORG-DEMO-REGION
 -OK-
 SUBSCHEMA=EMPSS01   SCHEMA=EMPSCHM    VER=0100    COL 001-080  LINE 0001 OF 0005
Te█                                                       R 41 C  2  TCPI0339
```

Keeping your system running

DMLO ‘query’ example

DMLO – Obtain Calc

1. Go to a record in DMLO (maybe Obtain first)
2. Overtype the ‘key’ field with the value you want to search for
3. Use OBTAIN CALC <record name>



Overtype what was originally here (5300) with the value you want to search for (3100)

```
 09/21/17.......................RECORD : DEPARTMENT......................13:29:27
 02 DEPT-ID-0410..............................N  3100
 02 DEPT-NAME-0410............................A  BLUE SKIES
    ==============+0032.......................A
 02 DEPT-HEAD-ID-0410.........................N  0321
 02 FILLER#001................................A* 000000



 DML/O R18.5 =========================================================== CA, INC.
 RECORD=DEPARTMENT       STATUS=0000   DBKEY=0000400081-0001 KEY0=0000000000-0000
 OBTAIN CALC DEPARTMENT
 -OK-
 SUBSCHEMA=EMPSS01   SCHEMA=EMPSCHM    VER=0100    COL 001-080  LINE 0001 OF 0005
Te█                             ⇧                         R  2 C 53  TCPI0339
```

Keeping your system running

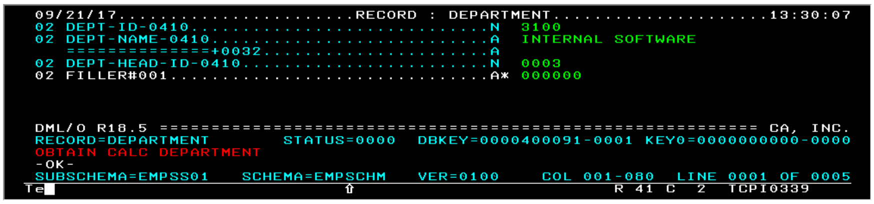
DMLO ‘query’ example

DMLO – Obtain Calc

This is a little scary since you are overtyping data in what looks like a full screen editor.

There are 2 safety valves –

1. You entered the session with READY MODE = Retrieval (can’t update in retrieval mode)
2. You are going to exit DMLO with a ROLLBACK command.



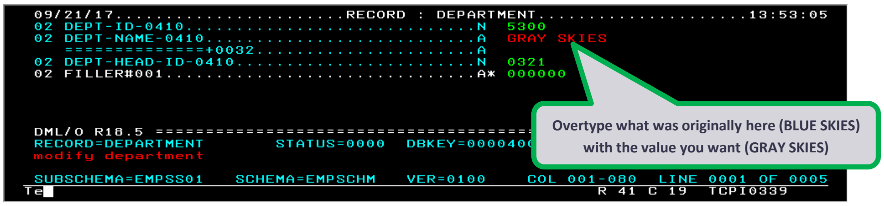
```
   09/21/17......................RECORD : DEPARTMENT....................13:30:07
   02 DEPT-ID-0410.............................N  3100
   02 DEPT-NAME-0410...........................A  INTERNAL SOFTWARE
      ==============+0032......................A
   02 DEPT-HEAD-ID-0410........................N  0003
   02 FILLER#001...............................A* 000000



   DML/O R18.5 ========================================================= CA, INC.
   RECORD=DEPARTMENT       STATUS=0000  DBKEY=0000400091-0001 KEY0=0000000000-0000
   OBTAIN CALC DEPARTMENT
   -OK-
   SUBSCHEMA=EMPSS01    SCHEMA=EMPSCHM    VER=0100    COL 001-080  LINE 0001 OF 0005
  Te                             ⇧                          R 41  C   2   TCPI0339
```

Keeping your system running

DMLO ‘query’ example

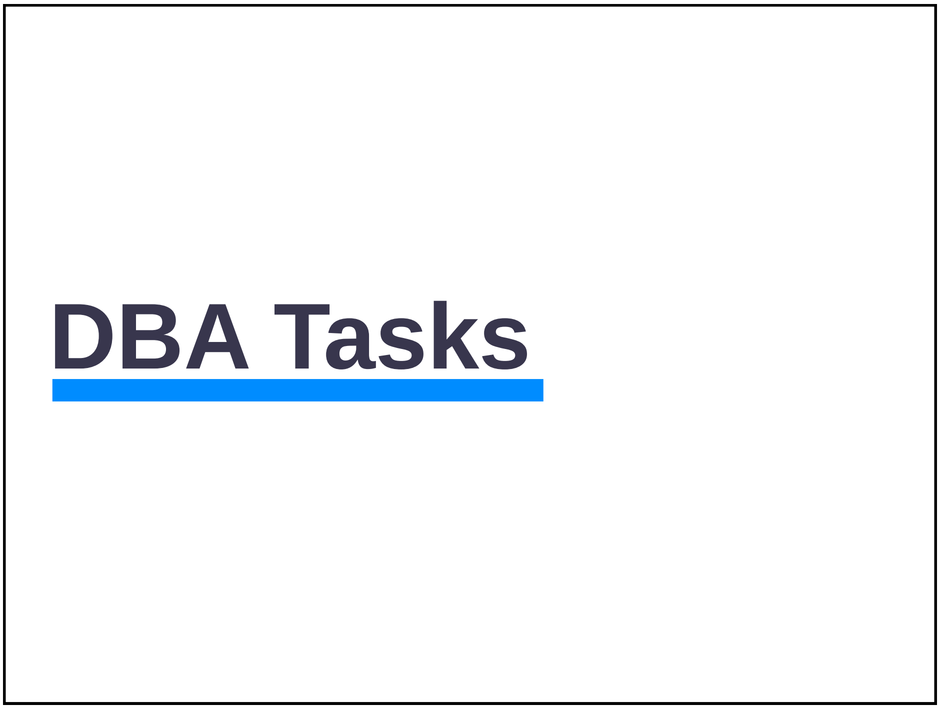
DMLO – Saving a change

1. Enter DMLO in Update mode – use ‘U’ on the DMLO ‘subschema’ panel
2. Then get to the record you want to change, make the edit, it will turn red, then type “MODIFY <record name>” to make it ‘take’
3. Upon exit the updates will be ready for commit or rollback. To Commit type FINISH. To rollback type ROLLBACK.

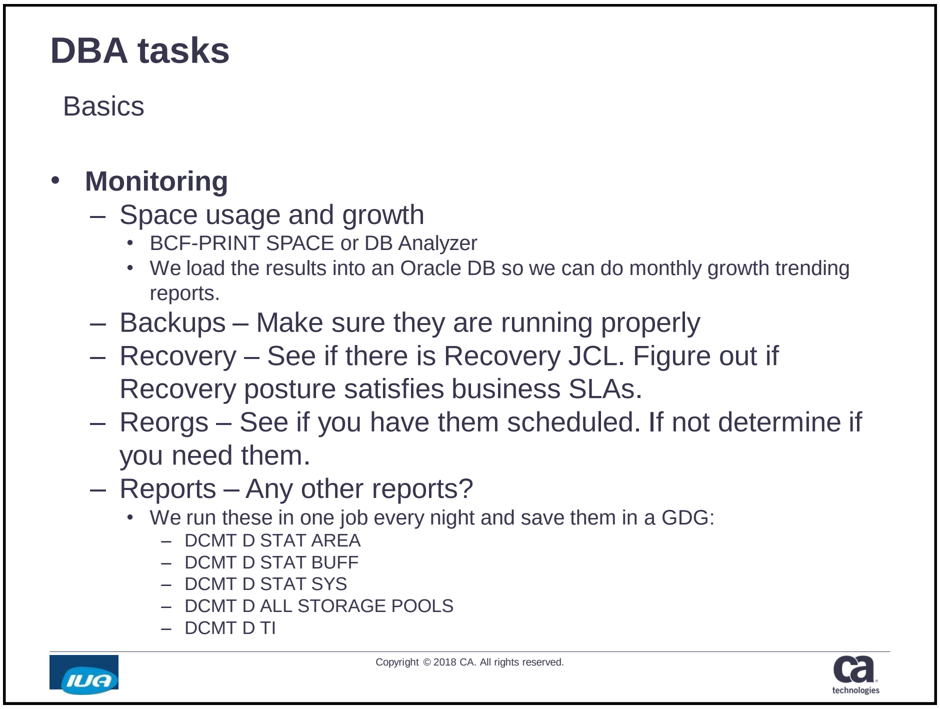


```
   09/21/17......................RECORD : DEPARTMENT....................13:53:05
   02 DEPT-ID-0410.............................N  5300
   02 DEPT-NAME-0410...........................A  GRAY SKIES
      ==============+0032......................A
   02 DEPT-HEAD-ID-0410........................N  0321
   02 FILLER#001...............................A* 000000



   DML/O R18.5 ====================================
   RECORD=DEPARTMENT       STATUS=0000  DBKEY=000040
   modify department

   SUBSCHEMA=EMPSS01    SCHEMA=EMPSCHM    VER=0100    COL 001-080  LINE 0001 OF 0005
  Te                                                        R 41  C 19   TCPI0339
```

Overtype what was originally here (BLUE SKIES) with the value you want (GRAY SKIES)



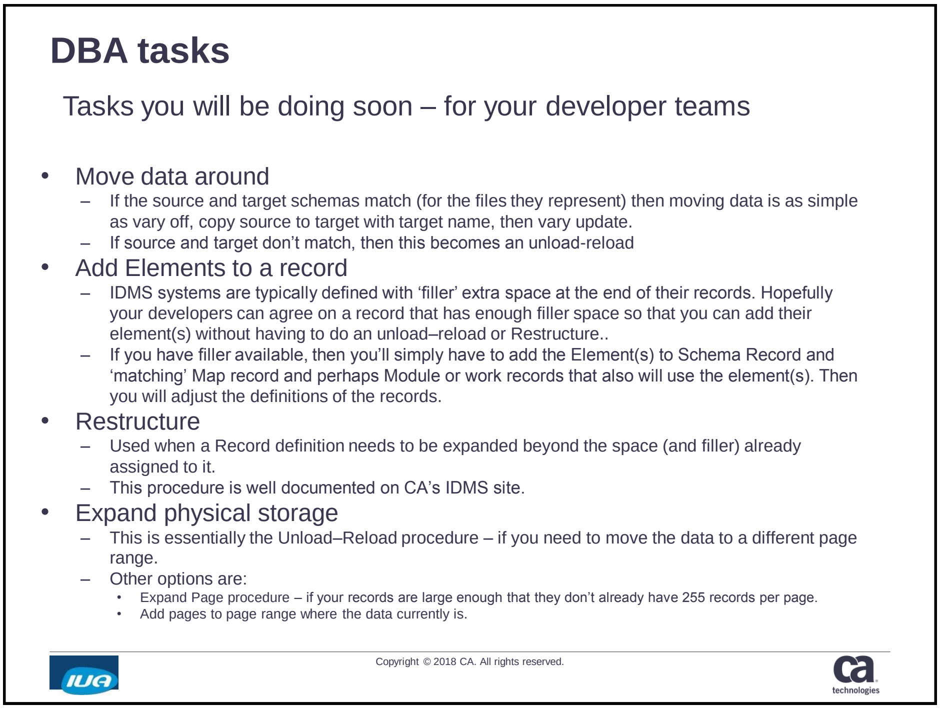
DBA Tasks



DBA tasks

Basics

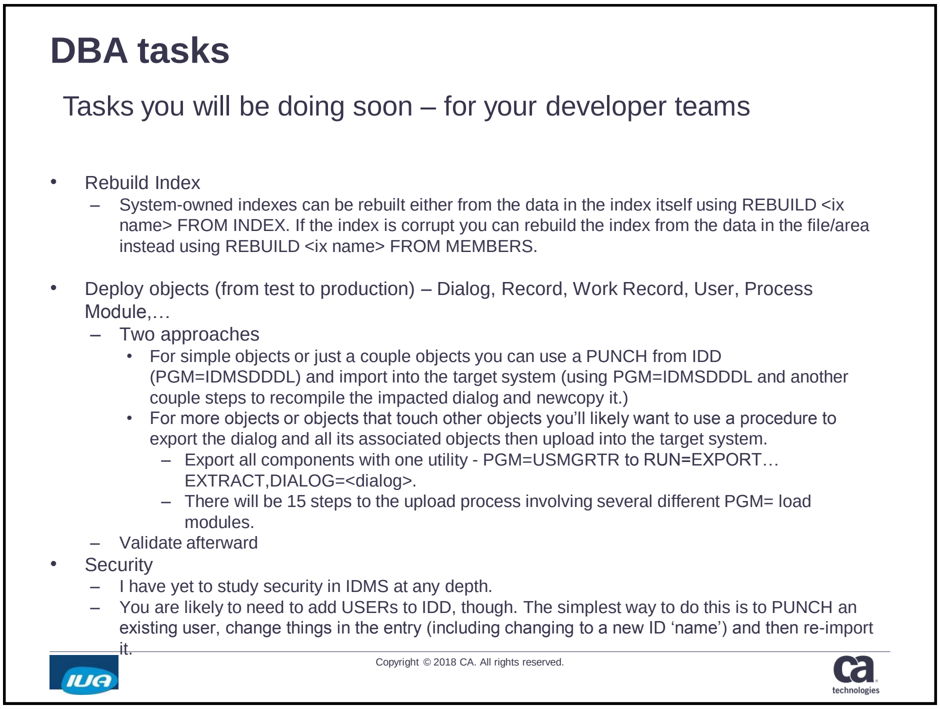
- **Monitoring**
  - Space usage and growth
    - BCF-PRINT SPACE or DB Analyzer
    - We load the results into an Oracle DB so we can do monthly growth trending reports.
  - Backups – Make sure they are running properly
  - Recovery – See if there is Recovery JCL. Figure out if Recovery posture satisfies business SLAs.
  - Reorgs – See if you have them scheduled. If not determine if you need them.
  - Reports – Any other reports?
    - We run these in one job every night and save them in a GDG:
      - DCMT D STAT AREA
      - DCMT D STAT BUFF
      - DCMT D STAT SYS
      - DCMT D ALL STORAGE POOLS
      - DCMT D TI



DBA tasks

Tasks you will be doing soon – for your developer teams

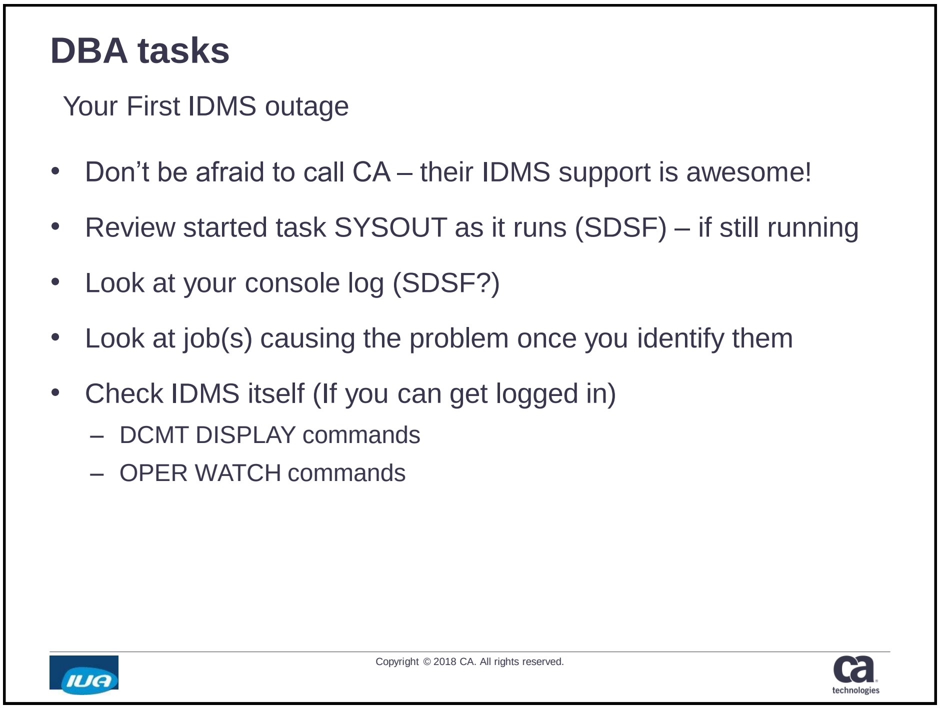
- Move data around
  - If the source and target schemas match (for the files they represent) then moving data is as simple as vary off, copy source to target with target name, then vary update.
  - If source and target don't match, then this becomes an unload-reload
- Add Elements to a record
  - IDMS systems are typically defined with 'filler' extra space at the end of their records. Hopefully your developers can agree on a record that has enough filler space so that you can add their element(s) without having to do an unload–reload or Restructure..
  - If you have filler available, then you'll simply have to add the Element(s) to Schema Record and 'matching' Map record and perhaps Module or work records that also will use the element(s). Then you will adjust the definitions of the records.
- Restructure
  - Used when a Record definition needs to be expanded beyond the space (and filler) already assigned to it.
  - This procedure is well documented on CA's IDMS site.
- Expand physical storage
  - This is essentially the Unload–Reload procedure – if you need to move the data to a different page range.
  - Other options are:
    - Expand Page procedure – if your records are large enough that they don't already have 255 records per page.
    - Add pages to page range where the data currently is.



DBA tasks

Tasks you will be doing soon – for your developer teams

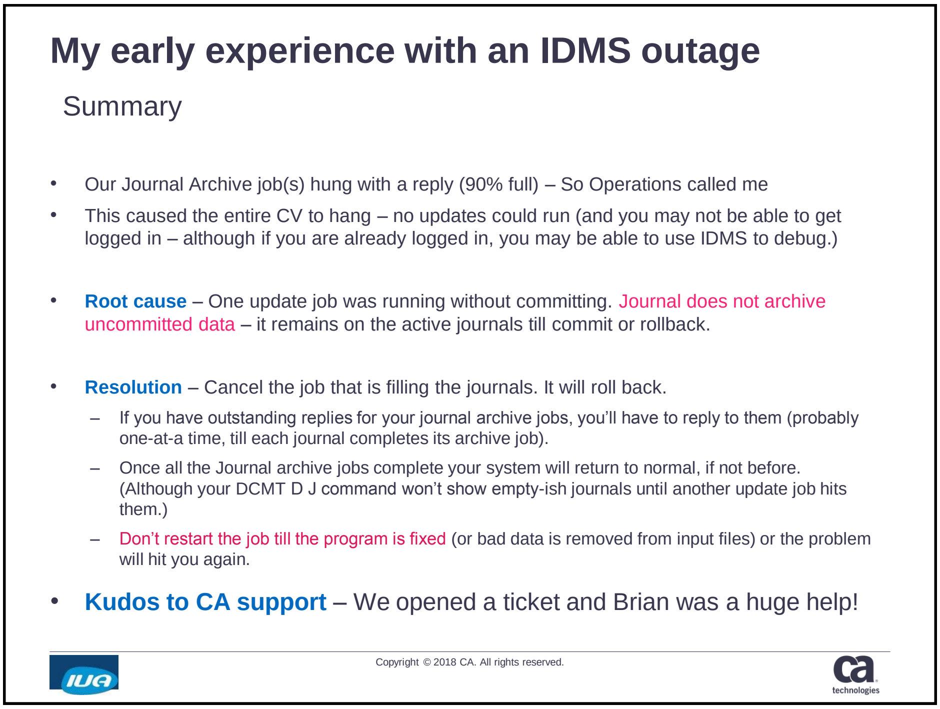
- Rebuild Index
  - System-owned indexes can be rebuilt either from the data in the index itself using REBUILD <ix name> FROM INDEX. If the index is corrupt you can rebuild the index from the data in the file/area instead using REBUILD <ix name> FROM MEMBERS.
- Deploy objects (from test to production) – Dialog, Record, Work Record, User, Process Module,…
  - Two approaches
    - For simple objects or just a couple objects you can use a PUNCH from IDD (PGM=IDMSDDDL) and import into the target system (using PGM=IDMSDDDL and another couple steps to recompile the impacted dialog and newcopy it.)
    - For more objects or objects that touch other objects you'll likely want to use a procedure to export the dialog and all its associated objects then upload into the target system.
      - Export all components with one utility - PGM=USMGRTR to RUN=EXPORT… EXTRACT,DIALOG=<dialog>.
      - There will be 15 steps to the upload process involving several different PGM= load modules.
  - Validate afterward
- Security
  - I have yet to study security in IDMS at any depth.
  - You are likely to need to add USERs to IDD, though. The simplest way to do this is to PUNCH an existing user, change things in the entry (including changing to a new ID 'name') and then re-import it.



DBA tasks

Your First IDMS outage

- Don’t be afraid to call CA – their IDMS support is awesome!
- Review started task SYSOUT as it runs (SDSF) – if still running
- Look at your console log (SDSF?)
- Look at job(s) causing the problem once you identify them
- Check IDMS itself (If you can get logged in)
  - DCMT DISPLAY commands
  - OPER WATCH commands



My early experience with an IDMS outage

Summary

- Our Journal Archive job(s) hung with a reply (90% full) – So Operations called me
- This caused the entire CV to hang – no updates could run (and you may not be able to get logged in – although if you are already logged in, you may be able to use IDMS to debug.)
- **Root cause** – One update job was running without committing. Journal does not archive uncommitted data – it remains on the active journals till commit or rollback.
- **Resolution** – Cancel the job that is filling the journals. It will roll back.
  - If you have outstanding replies for your journal archive jobs, you’ll have to reply to them (probably one-at-a time, till each journal completes its archive job).
  - Once all the Journal archive jobs complete your system will return to normal, if not before. (Although your DCMT D J command won’t show empty-ish journals until another update job hits them.)
  - Don’t restart the job till the program is fixed (or bad data is removed from input files) or the problem will hit you again.
- **Kudos to CA support** – We opened a ticket and Brian was a huge help!

My early experience with an IDMS outage

Diagnosis - IDMS

If you can get logged into IDMS

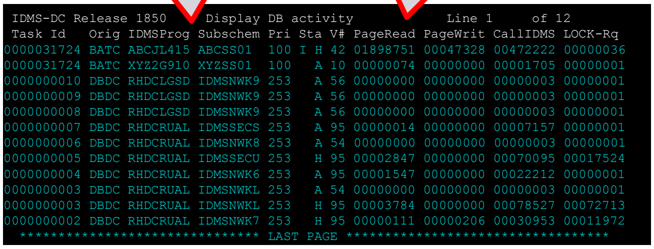
- Use OPER W DB and OPER W ACTIVE TASKS – both may indicate either the database or user/job that might be hogging the system
- Use DCMT DIS JOURNAL – see if the journals really are struggling.

My early experience with an IDMS outage

Symptoms of a CV that is about to break – OPER W DB

Although it doesn't name the job it, tells us the Subschema and Program name (load module). That should lead us to the Job in SDSF.

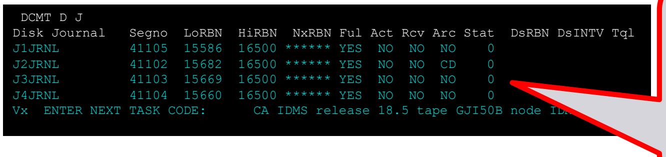
High PageRead, PageWrit and CallIDMS are indicators that this is a very busy program and may be the culprit.



```
 IDMS-DC Release 1850      Display DB activity            Line 1     of 12
 Task Id   Orig IDMSProg Subschem Pri Sta V# PageRead PageWrit CallIDMS LOCK-Rq
0000031724 BATC ABCJL415 ABCSS01  100 I H 42 01898751 00047328 00472222 00000036
0000031724 BATC XYZ2G910 XYZSS01  100   A 10 00000074 00000000 00001705 00000001
0000000010 DBDC RHDCLGSD IDMSNWK9 253   A 56 00000000 00000000 00000003 00000001
0000000009 DBDC RHDCLGSD IDMSNWK9 253   A 56 00000000 00000000 00000003 00000001
0000000008 DBDC RHDCLGSD IDMSNWK9 253   A 56 00000000 00000000 00000003 00000001
0000000007 DBDC RHDCRUAL IDMSSECS 253   A 95 00000014 00000000 00007157 00000001
0000000006 DBDC RHDCRUAL IDMSNWK8 253   A 54 00000000 00000000 00000003 00000001
0000000005 DBDC RHDCRUAL IDMSSECU 253   H 95 00002847 00000000 00070095 00017524
0000000004 DBDC RHDCRUAL IDMSNWK6 253   A 95 00001547 00000000 00022212 00000001
0000000003 DBDC RHDCRUAL IDMSNWKL 253   A 54 00000000 00000000 00000003 00000001
0000000003 DBDC RHDCRUAL IDMSNWKL 253   H 95 00003784 00000000 00078527 00072713
0000000002 DBDC RHDCRUAL IDMSNWK7 253   H 95 00000111 00000206 00030953 00011972
  ****************************** LAST PAGE *********************************
```

My early experience with an IDMS outage

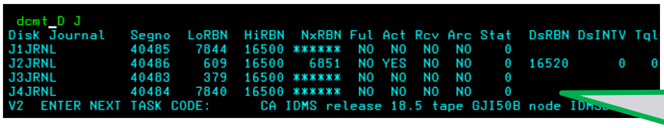
Symptoms of a CV that is about to break – DCMT DIS JOURNAL



```
 DCMT D J
Disk Journal   Segno  LoRBN  HiRBN  NxRBN Ful Act Rcv Arc Stat  DsRBN DsINTV Tql
J1JRNL         41105  15586  16500 ****** YES  NO  NO  NO    0
J2JRNL         41102  15682  16500 ****** YES  NO  NO  CD    0
J3JRNL         41103  15669  16500 ****** YES  NO  NO  NO    0
J4JRNL         41104  15660  16500 ****** YES  NO  NO  NO    0
Vx  ENTER NEXT TASK CODE:     CA IDMS release 18.5 tape GJI50B node ID
```

Couldn't get worse!

- LoRBN is very near the HiRBN.
- There is no NxRBN.
- All 4 journals indicate FUL=YES
- None of the Journals show as the ACTive one.



```
 dcmt_D J
Disk Journal   Segno  LoRBN  HiRBN  NxRBN Ful Act Rcv Arc Stat  DsRBN DsINTV Tql
J1JRNL         40485   7844  16500 ******  NO  NO  NO  NO    0
J2JRNL         40486    609  16500   6851  NO YES  NO  NO    0  16520      0   0
J3JRNL         40483    379  16500 ******  NO  NO  NO  NO    0
J4JRNL         40484   7840  16500 ******  NO  NO  NO  NO    0
V2  ENTER NEXT TASK CODE:      CA IDMS release 18.5 tape GJI50B node IDMS
```

Healthy Journals

- LoRBN not near HiRBN
- One NxRBN is the ACTive Journal
- None are FUL=YES

My early experience with an IDMS outage

Diagnosis - SDSF

If you can't get into IDMS:

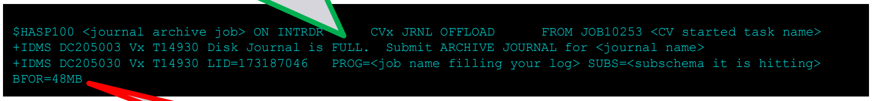
- Look in SDSF to see if you can find the job that is hanging. Large sysout, Long running, High CPU?
- In z/OS console log:
    - Journal Archive job will probably be issuing messages. Maybe an outstanding 'reply'.
    - You can see the messages and reply too, through SDSF z/OS system console log.
    - Look for messages like "Journal 90% full after condense."
- In CV started task 'job' on SDSF:
    - Your Journal filling up is NOT a problem (DC205003).
    - But DC205030 – is a big concern.
        - The BFOR=48MB indicates how much of the journal is consumed by this specific program.subschema's data.
    - You will likely see a number of messages like the above with increasing numbers in BFOR=xxMB. That will level off and probably stop increasing as the journal approaches 100% full.
    - Eventually you will get these really bad messages:
        - DC205013 - **All Disk Journals are UNAVAILABLE**
        - DC205004 - Journal is OFFLOADING/CONDENSING. Waiting for <archive job for first journal>
        - DC205014 - ARCHIVE JOURNAL NOT yet RUN against full Journal <archive job for second journal>
        - You'll get DC205014 for each of your Journals
        - DC205011 - **NO Journals are AVAILABLE.** IDMS Waiting

My early experience with an IDMS outage

Diagnosis – SDSF – CV Started Task

CV Started Task

- Journal FULL is not unusual. They always fill before the archive job runs.



```
$HASP100 <journal archive job> ON INTRDR      CVx JRNL OFFLOAD      FROM JOB10253 <CV started task name>
+IDMS DC205003 Vx T14930 Disk Journal is FULL.  Submit ARCHIVE JOURNAL for <journal name>
+IDMS DC205030 Vx T14930 LID=173187046   PROG=<job name filling your log> SUBS=<subschema it is hitting>
BFOR=48MB
```

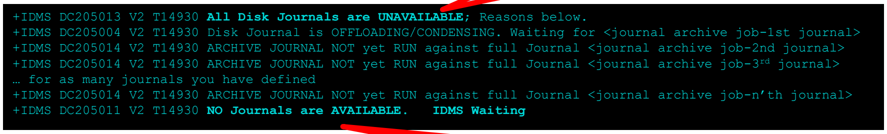
- These messages repeated with increasing numbers in BFOR=xxxMB is a BIG concern.

My early experience with an IDMS outage

Diagnosis - SDSF – CV Started Task

CV Started Task

- All Disk Journals are UNAVAILABLE



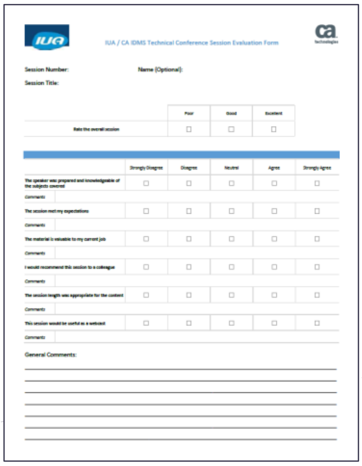
```
+IDMS DC205013 V2 T14930 All Disk Journals are UNAVAILABLE; Reasons below.
+IDMS DC205004 V2 T14930 Disk Journal is OFFLOADING/CONDENSING. Waiting for <journal archive job-1st journal>
+IDMS DC205014 V2 T14930 ARCHIVE JOURNAL NOT yet RUN against full Journal <journal archive job-2nd journal>
+IDMS DC205014 V2 T14930 ARCHIVE JOURNAL NOT yet RUN against full Journal <journal archive job-3rd journal>
… for as many journals you have defined
+IDMS DC205014 V2 T14930 ARCHIVE JOURNAL NOT yet RUN against full Journal <journal archive job-n’th journal>
+IDMS DC205011 V2 T14930 NO Journals are AVAILABLE.   IDMS Waiting
```

- NO Journals are AVAILABLE. IDMS Waiting

Questions & Answers

Please Complete a Session Evaluation Form

- The number for this session is **D03**
- After completing your session evaluation form, place it in the envelope at the front of the room



IUA / CA IDMS Technical Conference Session Evaluation Form

Session Number: Name (Optional):

Session Title:

	Poor	Good	Excellent
Rate the overall session	□	□	□

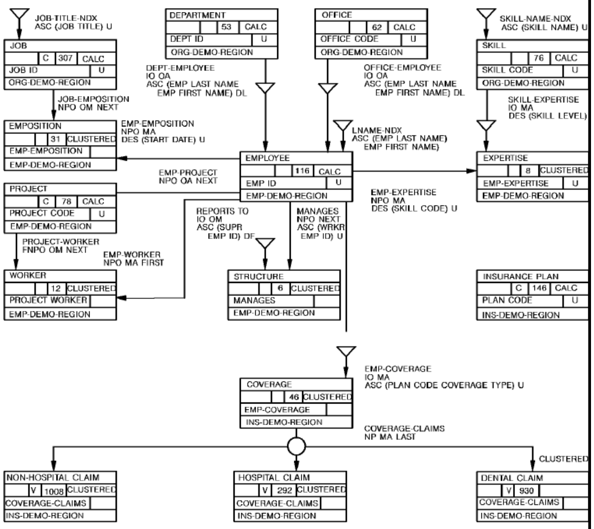
	Strongly Disagree	Disagree	Neutral	Agree	Strongly Agree
The speaker was prepared and knowledgeable of the subjects covered	□	□	□	□	□
Comments					
The session met my expectations	□	□	□	□	□
Comments					
The material is valuable to my current job	□	□	□	□	□
Comments					
I would recommend this session to a colleague	□	□	□	□	□
Comments					
The session length was appropriate for the content	□	□	□	□	□
Comments					
This session would be useful as a webcast	□	□	□	□	□
Comments					

General Comments:

Appendices

Appendix 1

IDMS Sample (Demo) Database Diagram



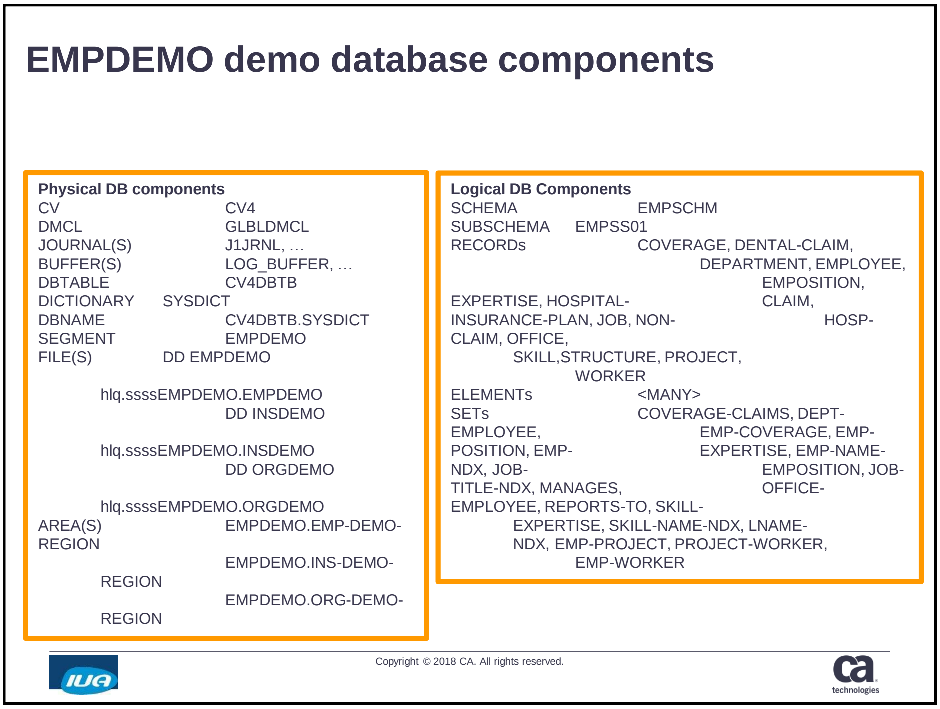
Obtain first department within org-demo-region

Obtain first employee within dept-employee

Obtain next employee WITHIN DEPT-EMPLOYEE

Obtain next emposition within emp-emposition

OBTAIN OWNER WITHIN DEPT-EMPLOYEE



EMPDEMO demo database components

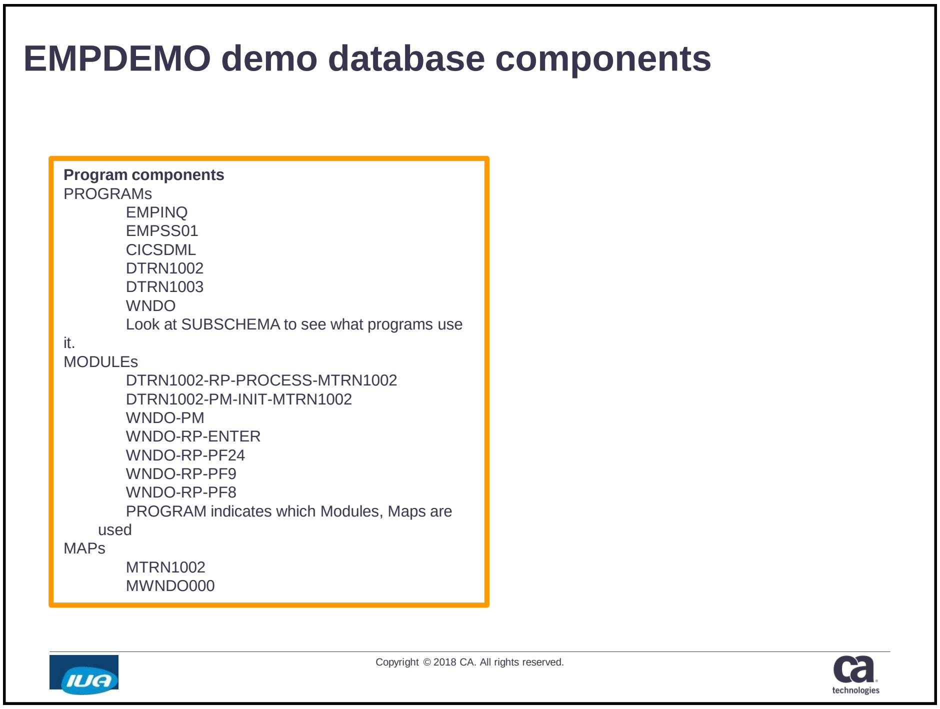
Physical DB components
CV CV4
DMCL GLBLDMCL
JOURNAL(S) J1JRNL, …
BUFFER(S) LOG_BUFFER, …
DBTABLE CV4DBTB
DICTIONARY SYSDICT
DBNAME CV4DBTB.SYSDICT
SEGMENT EMPDEMO
FILE(S) DD EMPDEMO

hlq.ssssEMPDEMO.EMPDEMO
DD INSDEMO

hlq.ssssEMPDEMO.INSDEMO
DD ORGDEMO

hlq.ssssEMPDEMO.ORGDEMO
AREA(S) EMPDEMO.EMP-DEMO-
REGION
EMPDEMO.INS-DEMO-
REGION
EMPDEMO.ORG-DEMO-
REGION

Logical DB Components
SCHEMA EMPSCHM
SUBSCHEMA EMPSS01
RECORDs COVERAGE, DENTAL-CLAIM,
DEPARTMENT, EMPLOYEE,
EMPOSITION,
EXPERTISE, HOSPITAL- CLAIM,
INSURANCE-PLAN, JOB, NON- HOSP-
CLAIM, OFFICE,
SKILL,STRUCTURE, PROJECT,
WORKER
ELEMENTs <MANY>
SETs COVERAGE-CLAIMS, DEPT-
EMPLOYEE, EMP-COVERAGE, EMP-
POSITION, EMP- EXPERTISE, EMP-NAME-
NDX, JOB- EMPOSITION, JOB-
TITLE-NDX, MANAGES, OFFICE-
EMPLOYEE, REPORTS-TO, SKILL-
EXPERTISE, SKILL-NAME-NDX, LNAME-
NDX, EMP-PROJECT, PROJECT-WORKER,
EMP-WORKER



EMPDEMO demo database components

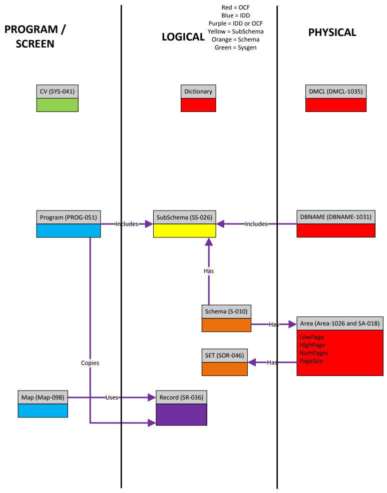
Program components
PROGRAMs
EMPINQ
EMPSS01
CICSDML
DTRN1002
DTRN1003
WNDO
Look at SUBSCHEMA to see what programs use it.
MODULEs
DTRN1002-RP-PROCESS-MTRN1002
DTRN1002-PM-INIT-MTRN1002
WNDO-PM
WNDO-RP-ENTER
WNDO-RP-PF24
WNDO-RP-PF9
WNDO-RP-PF8
PROGRAM indicates which Modules, Maps are used
MAPs
MTRN1002
MWNDO000

Appendix 2

IDMS Dictionary

Entity Relat. Diagram

– Super Simplified



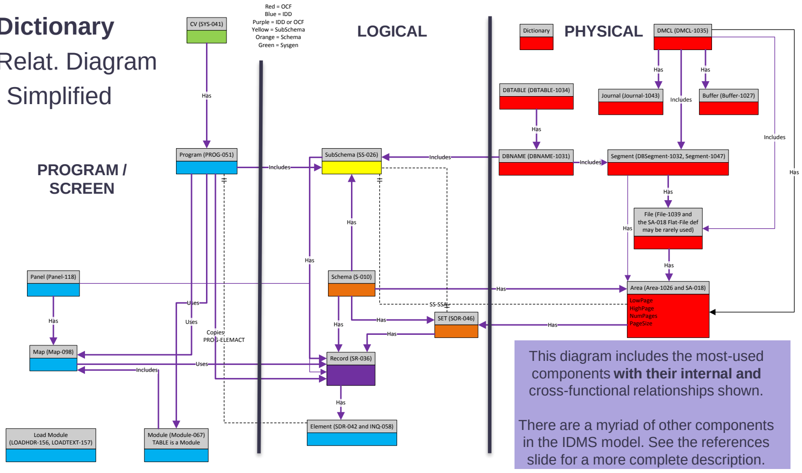
This diagram includes **only** the most-used components with their cross-functional relationships shown.

Components that don't have cross-functional relationships are not shown.

IDMS Dictionary

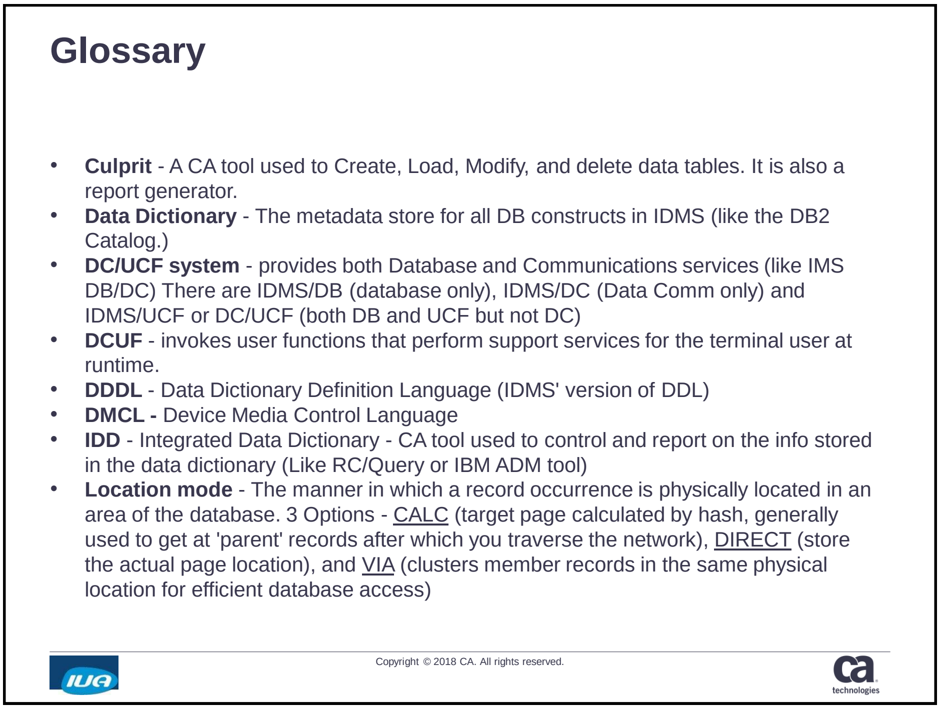
Entity Relat. Diagram

– Less Simplified



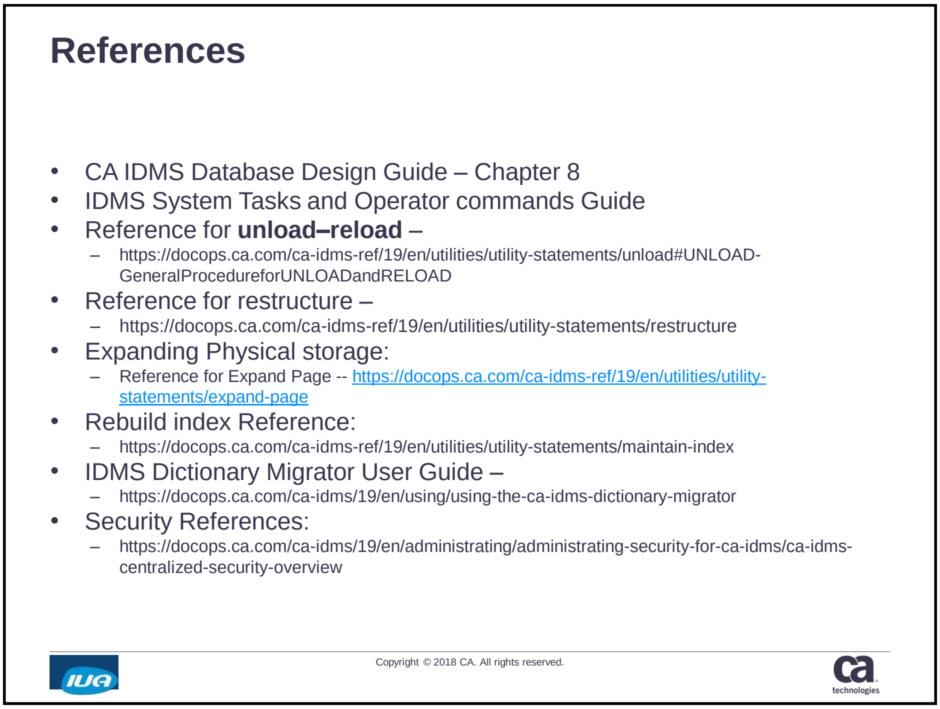
This diagram includes the most-used components **with their internal and** cross-functional relationships shown.

There are a myriad of other components in the IDMS model. See the references slide for a more complete description.



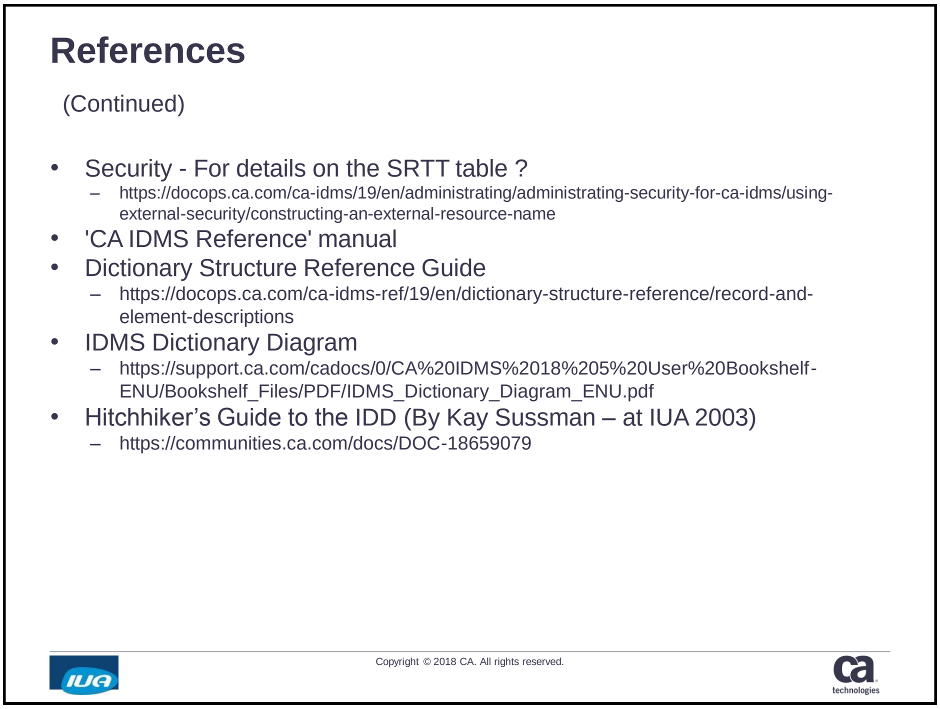
Glossary

- **Culprit** - A CA tool used to Create, Load, Modify, and delete data tables. It is also a report generator.
- **Data Dictionary** - The metadata store for all DB constructs in IDMS (like the DB2 Catalog.)
- **DC/UCF system** - provides both Database and Communications services (like IMS DB/DC) There are IDMS/DB (database only), IDMS/DC (Data Comm only) and IDMS/UCF or DC/UCF (both DB and UCF but not DC)
- **DCUF** - invokes user functions that perform support services for the terminal user at runtime.
- **DDDL** - Data Dictionary Definition Language (IDMS' version of DDL)
- **DMCL -** Device Media Control Language
- **IDD** - Integrated Data Dictionary - CA tool used to control and report on the info stored in the data dictionary (Like RC/Query or IBM ADM tool)
- **Location mode** - The manner in which a record occurrence is physically located in an area of the database. 3 Options - CALC (target page calculated by hash, generally used to get at 'parent' records after which you traverse the network), DIRECT (store the actual page location), and VIA (clusters member records in the same physical location for efficient database access)



References

- CA IDMS Database Design Guide – Chapter 8
- IDMS System Tasks and Operator commands Guide
- Reference for **unload–reload** –
  - https://docops.ca.com/ca-idms-ref/19/en/utilities/utility-statements/unload#UNLOAD-GeneralProcedureforUNLOADandRELOAD
- Reference for restructure –
  - https://docops.ca.com/ca-idms-ref/19/en/utilities/utility-statements/restructure
- Expanding Physical storage:
  - Reference for Expand Page -- https://docops.ca.com/ca-idms-ref/19/en/utilities/utility-statements/expand-page
- Rebuild index Reference:
  - https://docops.ca.com/ca-idms-ref/19/en/utilities/utility-statements/maintain-index
- IDMS Dictionary Migrator User Guide –
  - https://docops.ca.com/ca-idms/19/en/using/using-the-ca-idms-dictionary-migrator
- Security References:
  - https://docops.ca.com/ca-idms/19/en/administrating/administrating-security-for-ca-idms/ca-idms-centralized-security-overview



References

(Continued)

- Security - For details on the SRTT table ?
  - https://docops.ca.com/ca-idms/19/en/administrating/administrating-security-for-ca-idms/using-external-security/constructing-an-external-resource-name
- 'CA IDMS Reference' manual
- Dictionary Structure Reference Guide
  - https://docops.ca.com/ca-idms-ref/19/en/dictionary-structure-reference/record-and-element-descriptions
- IDMS Dictionary Diagram
  - https://support.ca.com/cadocs/0/CA%20IDMS%2018%205%20User%20Bookshelf-ENU/Bookshelf_Files/PDF/IDMS_Dictionary_Diagram_ENU.pdf
- Hitchhiker’s Guide to the IDD (By Kay Sussman – at IUA 2003)
  - https://communities.ca.com/docs/DOC-18659079