

JAVA and SQL in IMS applications

- How to write Java applications in IDz to access your IMS data using SQL.

Radek Mrvec, CA Technologies, 13.9.2018



Disclaimer

- Certain information in this presentation may outline CA's general product direction. This presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. This presentation is based on current information and resource allocations as of May 1, 2018 and is **subject to change or withdrawal by CA at any time without notice. The development, release and timing of any features or functionality described in this presentation remain at CA's sole discretion.**
- Notwithstanding anything in this presentation to the contrary, upon the general availability of any future CA product release referenced in this presentation, CA may make such release available to new licensees in the form of a regularly scheduled major product release. Such release may be made available to licensees of the product who are active subscribers to CA maintenance and support, on a when and if-available basis. The information in this presentation is not deemed to be incorporated into any contract.
- Copyright © 2018 CA. All rights reserved. All trademarks, trade names, service marks and logos referenced herein belong to their respective companies.
- **THIS PRESENTATION IS FOR YOUR INFORMATIONAL PURPOSES ONLY.** CA assumes no responsibility for the accuracy or completeness of the information. TO THE EXTENT PERMITTED BY APPLICABLE LAW, CA PROVIDES THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. **In no event will CA be liable for any loss or damage, direct or indirect, in connection with this presentation, including, without limitation, lost profits, lost investment, business interruption, goodwill, or lost data, even if CA is expressly advised in advance of the possibility of such damages.**

Agenda

- 1 Who am I ?
- 2 What is it about ?
- 3 Time to prepare the environment
- 4 Is it all ?

Agenda

- 5 Prepare playground
- 6 Leaving green screen
- 7 Welcome to the SQL world
- 8 Let's do java

Who am I

Let me introduce myself...

What is it about?

What is this presentation about? How will it help me?

Time to prepare the environment

What needs to be set before we start?

Minimum required steps

IMS Catalog

Common Service
Layer (SCI)

IMS Connect (ICON)

Open Database
Manager (ODBM)

IMS Catalog and Managed ACBs

- Configuration PROCLIB member DFSDFxxx.

```
<SECTION=CATALOG>  
CATALOG=Y /*Enable IMS catalog*/  
ALIAS=DFSC /*Use standard catalog prefix DFSC*/  
RETENTION=(INSTANCES=5,DAYS=365) /*Retention criteria*/  
ACBMGMT=CATALOG /*Managed ACBs*/
```

COMMON SERVICE LAYER (SCI)

- Configuration PROCLIB member DFSDFxxx.

```
<SECTION=COMMON_SERVICE_LAYER>
ACBSHR=Y                /* Share ACB libraries */
CMDSEC=N                /* No cmd authorization checking*/
IMSPLEX=PLEX1          /* IMSplex name */
OLC=GLOBAL              /* GLOBAL online change */
OLCSTAT=IMSTESTS.IMS01.OLCSTAT /* OLCSTAT data set name */
MODBLKS=DYN            /* DRD ENABLED; OLC DISABLED */
PLEXPARM=( )           /* GLOBAL resource status */
UOM=MTO                /* Unsolicited output message support */
```

COMMON SERVICE LAYER (SCI)

- Configuration PROCLIB member CSLSIxxx.

```
ARMRST=Y          /* ARM should restart SCI on failure */
IMSPLEX(NAME=PLEX1) /* IMSplex name (CSLPLEX1) */
SCINAME=SCI1      /* SCI name (SCIID = SCI1SC) */
```

IMS CONNECT (ICON)

- Configuration PROCLIB member HWSCFGxx

```
HWS (ID=HWS1,RACF=N,XIBAREA=20)
TCPIP (HOSTNAME=MVSTCPIP,RACFID=RACFID,MAXSOC=2000)
ODACCESS (DRDAPORT=(ID=1111,KEEPAV=5,PORTTMOT=50),
IMSPLEX=(MEMBER=HWS1,TMEMBER=PLEX1),ODBMAUTOCONN=Y)
DATASTORE (ID=SOCKEYE,MEMBER=COHO,TMEMBER=CHINOOK,GROUP=SALMON)
```

OPEN DATABASE MANAGER (ODBM)

- Configuration PROCLIB MEMBER CSLDIxx

```

ARMRST=N           /* ARM should not restart ODBM on failure */
ODBMNAME=ODBM1     /* ODBM Name (ODBMID = ODBM1OD)          */
IMSPLEX(NAME=PLEX1) /* IMSplex Name (CSLPLEX1)              */
ODBMCFG=000        /* Configuration member (CSLDC000)       */
RRS=Y              /* RRS is the ODBM sync coordinator     */
  
```

OPEN DATABASE MANAGER (ODBM)

- Configuration PROCLIB member CSLDCxx

```

<SECTION=GLOBAL_DATASTORE_CONFIGURATION>
IDRETRY=5                /* Retry connection 5 times before quit */
MAXTHRDS=10             /* 10 threads max to any IMS Datastore */
TIMER=30                /* 30 seconds between ID retry attempts */
FPBUF=10               /* 10 DEDB buffers per thread */
FPBOF=10               /* 10 Overflow buffers per thread */
CNBA=200                /* (FPBUF+FPBOF)*MAXTHRDS <= CNBA */
/*****
/* Define DATASTORE properties for ODBM01 */
*****/
<SECTION=LOCAL_DATASTORE_CONFIGURATION>
ODBM (NAME=ODBM01,      /* Define parms for ODBM01 */
      DATASTORE (NAME=IMS1, /* IMSID on LPAR A */
                  ALIAS (NAME=IO1A, NAME=IO1B), /* Names for APPL sets 1 & 2 */
                  FPBUF=0, FPBOF=0, CNBA=0) /* No FastPath on this IMS */
      )
)
  
```

Is it all ?

We are ready now, but...

Other recommendations

- 1 OPERATION MANAGER (OM)
- 2 RESOURCE MANAGER (RM)
- 3 REPOSITORY SERVER (RS)
- 4 DYNAMIC RESOURCE DEFINITION (DRD)

Prepare playground

Create DBD and PSB for our Java Application

DBD

```
DBD NAME=ORDRMR,ACCESS=(HDAM,OSAM),          X
      RMNAME=(DFSHDC40,2,5)
DATASET DD1=ORDRMR,SIZE=2048
SEGM  NAME=XXORDER,PARENT=0,BYTES=8,PTR=(T)
FIELD NAME=(ID,SEQ,U),BYTES=4,START=1,DATATYPE=INT
FIELD NAME=ORDID,BYTES=4,START=5,DATATYPE=INT
SEGM  NAME=XXCUST,PARENT=XXORDER,BYTES=28,PTR=(T)
FIELD NAME=(ID,SEQ,U),BYTES=4,START=1,DATATYPE=INT
FIELD NAME=NAME,BYTES=24,START=5,TYPE=C
SEGM  NAME=XXPROD,PARENT=XXORDER,BYTES=34,PTR=(T)
FIELD NAME=(ID,SEQ,U),BYTES=4,START=1,DATATYPE=INT
FIELD NAME=NAME,BYTES=24,START=5,TYPE=C
FIELD NAME=PRICE,BYTES=6,START=29,DATATYPE=DECIMAL(10,2)
SEGM  NAME=XXSTORE,PARENT=XXORDER,BYTES=24,PTR=(T)
FIELD NAME=(ID,SEQ,U),BYTES=4,START=1,DATATYPE=INT
FIELD NAME=NAME,BYTES=20,START=5,TYPE=C
SEGM  NAME=XXSALES,PARENT=XXSTORE,BYTES=24,PTR=(T)
FIELD NAME=(ID,SEQ,U),BYTES=4,START=1,DATATYPE=INT
FIELD NAME=NAME,BYTES=20,START=5,TYPE=C
DBDGEN
FINISH
END
```

PSB

```
PORDMRG PCB TYPE=DB,DBDNAME=ORDRMR,KEYLEN=12,PROCOPT=A
*
* - SEGMENT ACCESIBLE BY PROGRAM
  SENSEG NAME=XXORDER,PARENT=0
  SENSEG NAME=XXCUST,PARENT=XXORDER
  SENSEG NAME=XXPROD,PARENT=XXORDER
  SENSEG NAME=XXSTORE,PARENT=XXORDER
  SENSEG NAME=XXSALES,PARENT=XXSTORE
*
* - WILL BE USED FROM ASSEMBLER, NAME IS PORDMRG, WILL HAVE ALWAYS I/O
  PSBGEN LANG=ASSEM,PSBNAME=PORDMRG,CMPAT=YES
  END
```

I don't want to write DBD or PSB

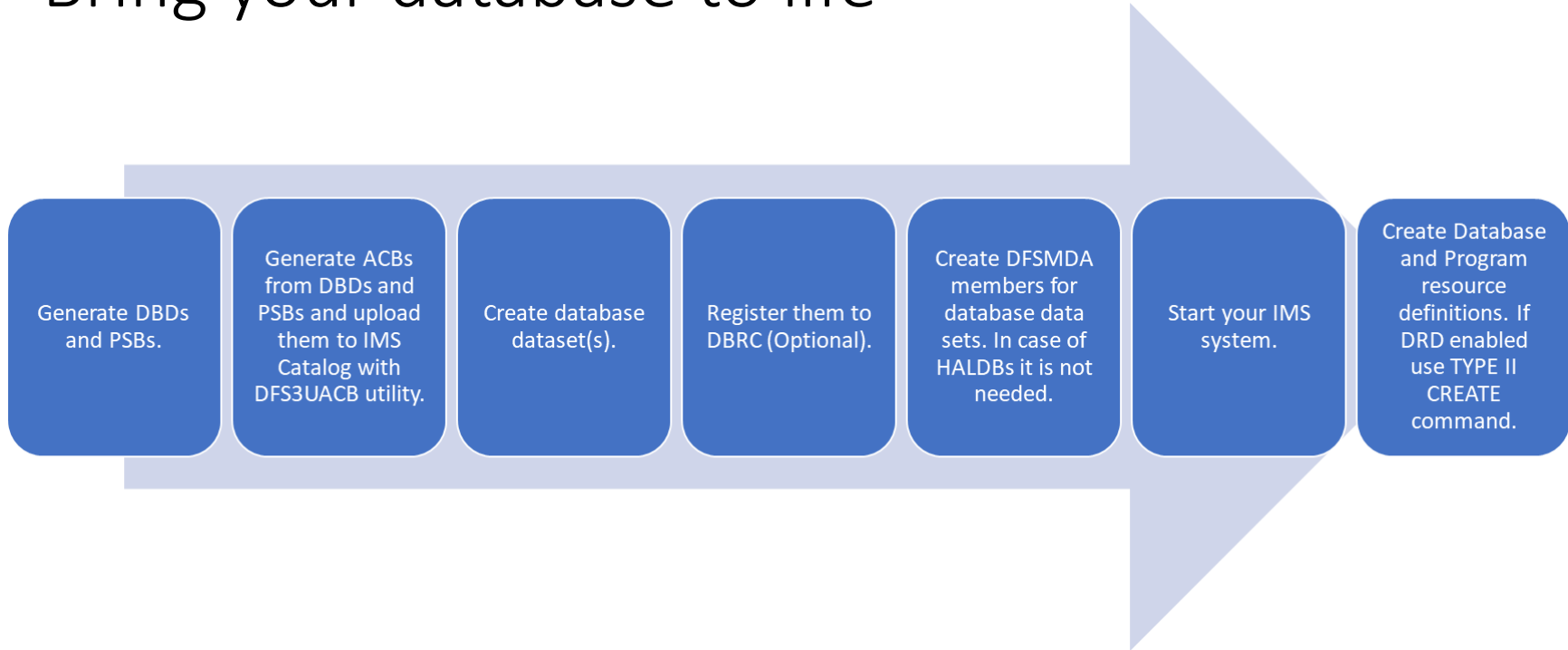
DDL For
creating DBD

- CREATE DATABASE
SKILLINV ACCESS
HDAM VSAM RMNAME
(DFSHDC40 RMANCH 20
RMRBN 500 RMBYTES 824)

DDL for
creating Root
segment

- CREATE TABLE
A INTERNALNAME A
MAXBYTES 48 MINBYTES 27
IN DATABASE SKILLINV

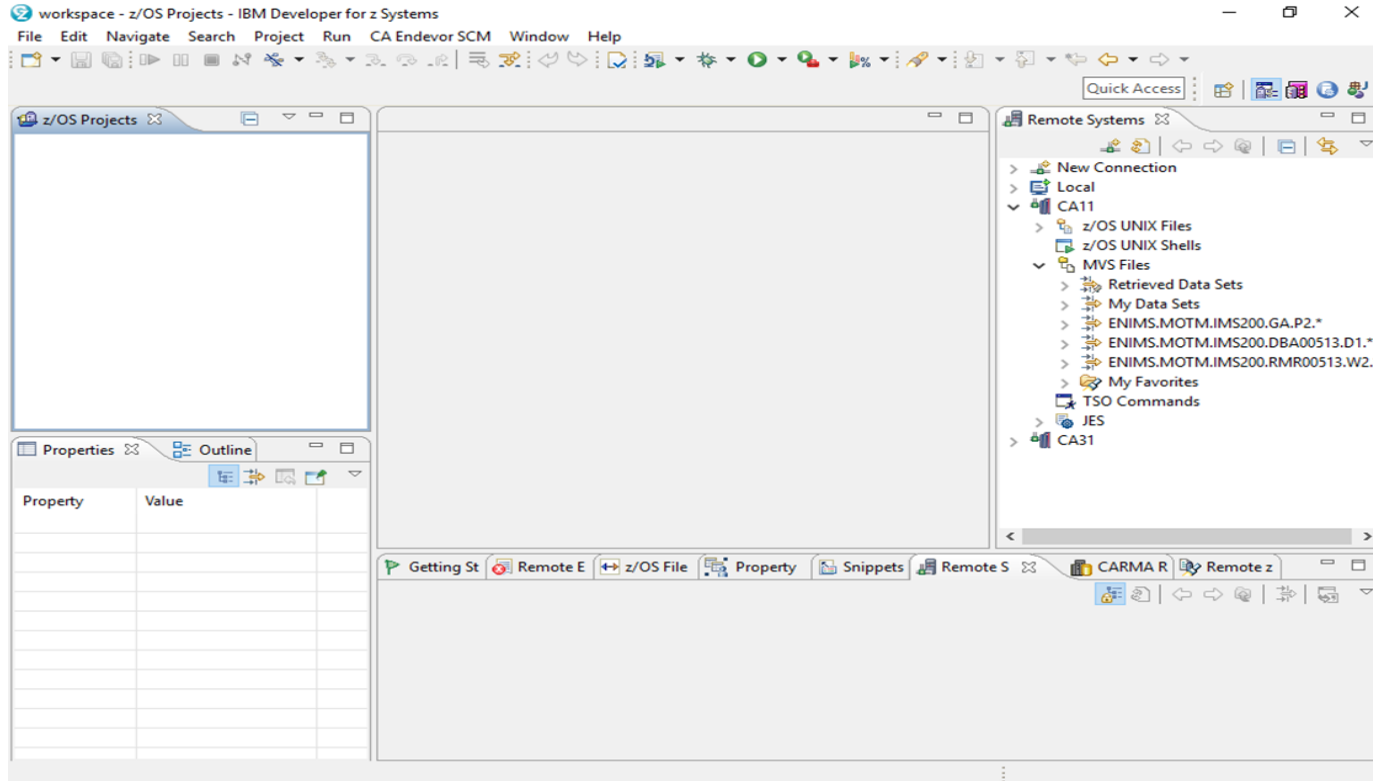
Bring your database to life



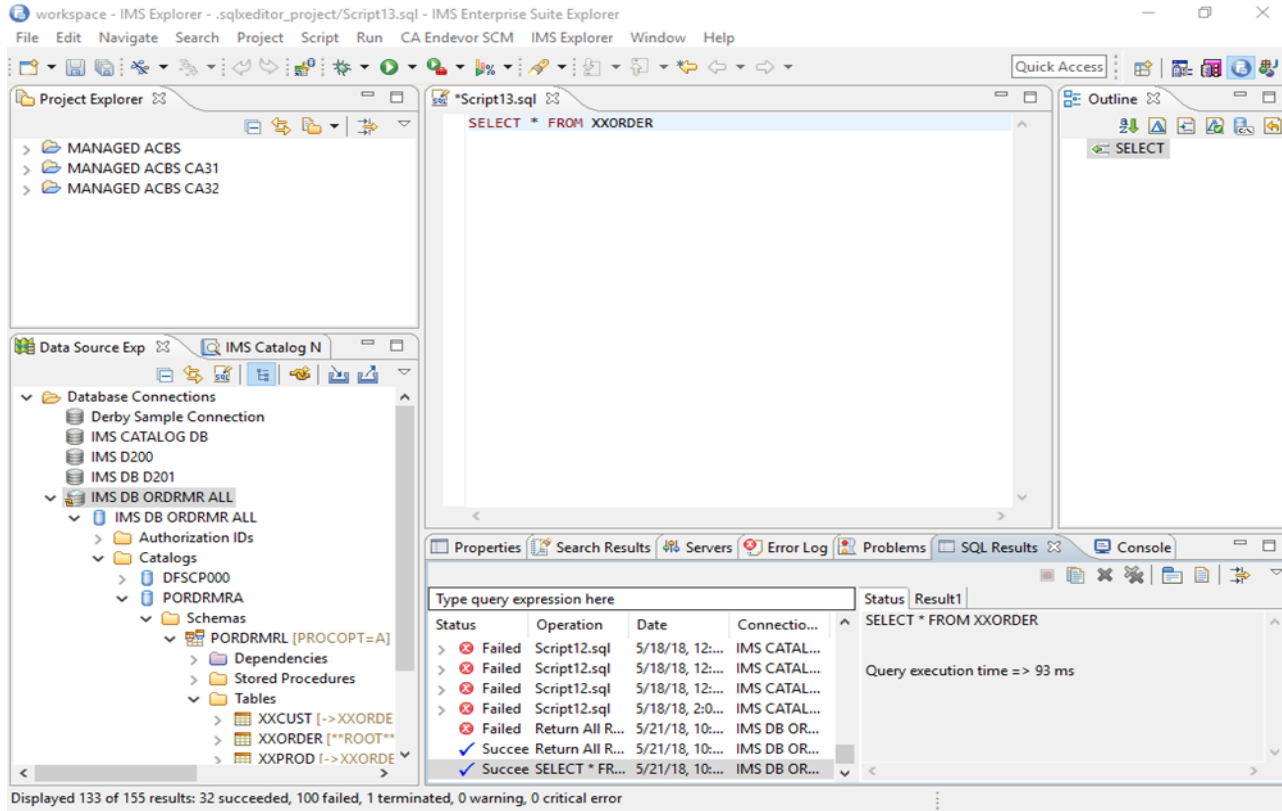
Leaving green screen

Time to try something new.

IBM Developer for z Systems



IMS Explorer for Development



The screenshot displays the IMS Explorer for Development interface. The top menu bar includes File, Edit, Navigate, Search, Project, Script, Run, CA Endeavor SCM, IMS Explorer, Window, and Help. The main workspace is divided into several panes:

- Project Explorer:** Shows a tree view of project folders: MANAGED ACBS, MANAGED ACBS CA31, and MANAGED ACBS CA32.
- Data Source Exp:** Shows a tree view of database connections and schemas. The selected connection is 'IMS DB ORDRMR ALL', which includes sub-nodes for Authorization IDs, Catalogs, Schemas, and Tables. The 'Tables' node is expanded, showing 'XXCUST [->XXORDE]', 'XXORDER [**ROOT**]', and 'XXPROD [->XXORDE]'.
- SQL Editor:** Contains a script named 'Script13.sql' with the query: `SELECT * FROM XXORDER`.
- SQL Results Console:** Displays the execution results of the query. It shows a table with columns: Status, Operation, Date, and Connection. The results are as follows:

Status	Operation	Date	Connection
Failed	Script12.sql	5/18/18, 12:...	IMS CATAL...
Failed	Script12.sql	5/18/18, 12:...	IMS CATAL...
Failed	Script12.sql	5/18/18, 12:...	IMS CATAL...
Failed	Script12.sql	5/18/18, 2:0...	IMS CATAL...
Failed	Return All R...	5/21/18, 10:...	IMS DB OR...
Succeeded	Return All R...	5/21/18, 10:...	IMS DB OR...
Succeeded	SELECT * FR...	5/21/18, 10:...	IMS DB OR...

At the bottom of the console, it states: "Displayed 133 of 155 results: 32 succeeded, 100 failed, 1 terminated, 0 warning, 0 critical error".

Connect to your database

Driver Properties

Drivers: IMS V14 Universal JDBC Driver Default

Properties

General Tracing Optional

*Connection name: IMS DB ORDRMR ALL

*Host: ca11

*Port number: 6667

User name:

Password:

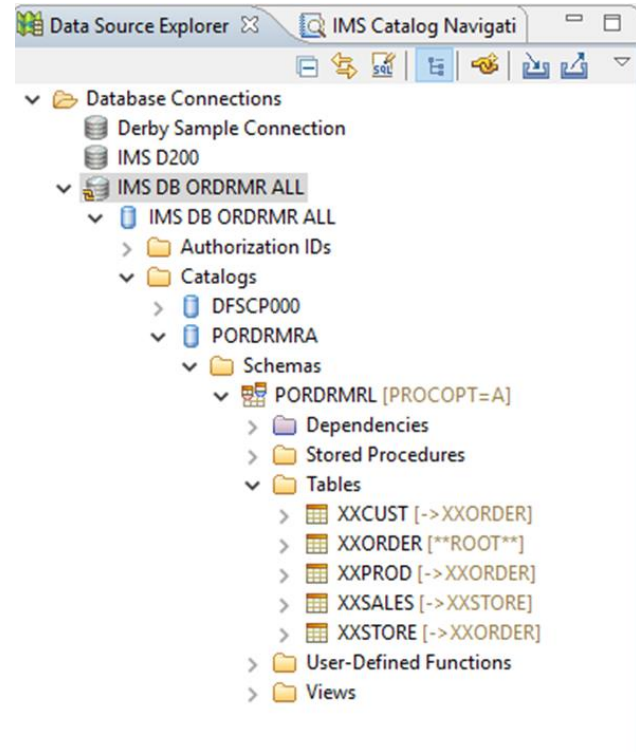
Save password

Default schema:

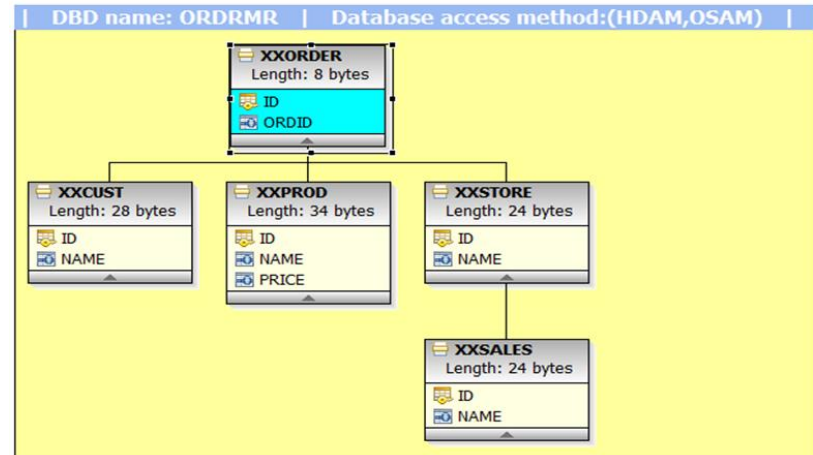
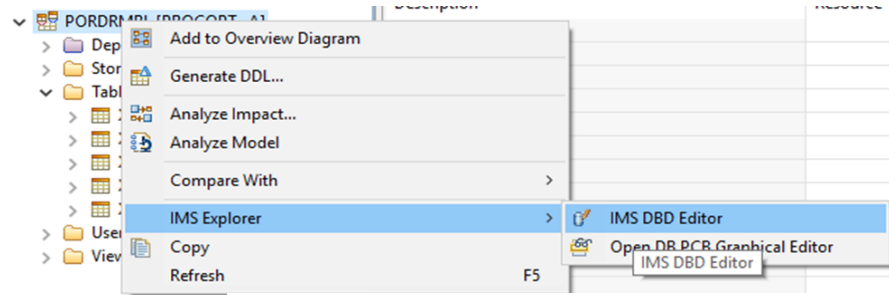
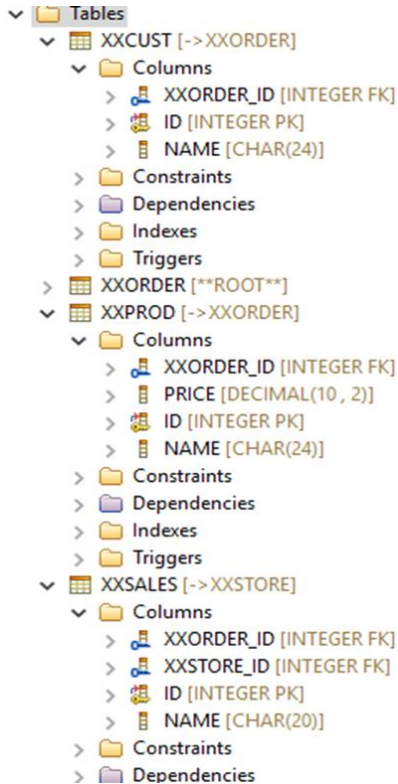
Metadata source: IMS Catalog

PSB: PORDRMRA

URL: jdbc:ims://ca11:6667/PORDRMRA:dpsbOnCommit=true;treatInvalidDecimalAsNull=true;fetchSize=0;flattenTables=true;



Connect to your database



Connect to your database

Length: 34 Bytes

Map name: Case name:

FIELD statement

Add or edit a physical DBD field for a segment by defining or modifying the FIELD statement. [Learn more...](#)

Name	Alias	Parent	Field type	Starting p...	Length	Data type	Physical d...	Redefines ...
ID	ID		IMS field	S=1	4	INT	INT	
NAME	NAME		IMS field	S=5	24	CHAR	CHAR	
PRICE	PRICE		IMS field	S=29	6	DECIMAL	PACKEDD...	

Welcome to the SQL world

How can IMS be relational?

Welcome in SQL world

```

S 111111111 2ORDMR
U -----C-----T----- (12345678OP----)---T-----
L   ISRT XXORDER
L   DATA 000118001
U
L   ISRT XXORDER
L     XXCUST
L   DATA 0001JAMES
U
L   ISRT XXORDER
L     XXPROD
L   DATA 0001HAMBURGER      010.50
U
L   ISRT XXORDER
L     XXSTORE
L   DATA 0001CHICAGO
U
L   ISRT XXORDER
L     XXSTORE
L     XXSALES
L   DATA 0001JOHN
  
```

Welcome in SQL world

```
INSERT INTO XXORDER (ID, ORDID) VALUES (1, 18001);
```

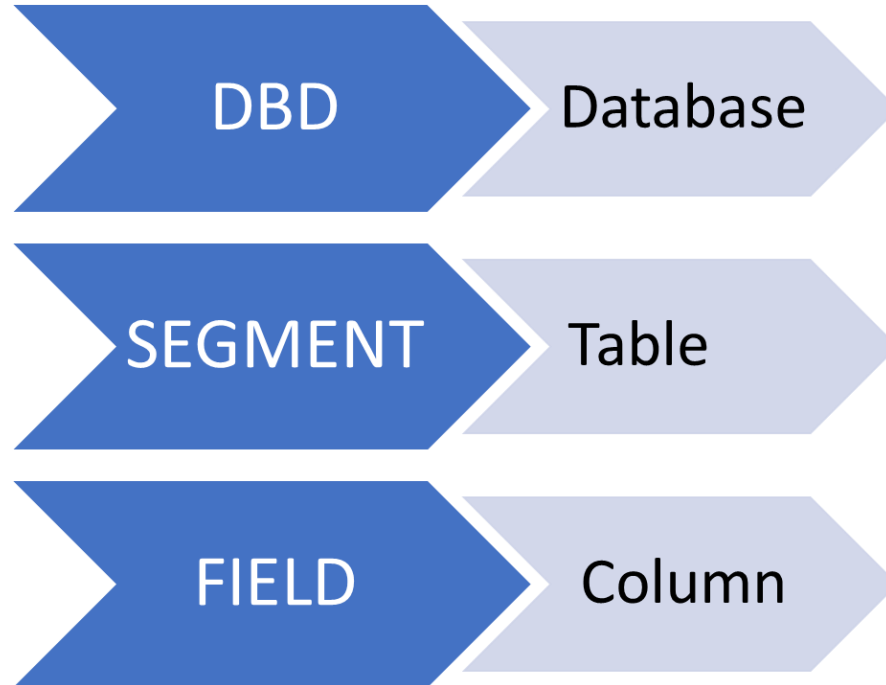
```
INSERT INTO XXCUST (ID, NAME, XXORDER_ID) VALUES (1,'JAMES',1);
```

```
INSERT INTO XXPROD (ID, PRICE, NAME, XXORDER_ID) VALUES (1,'10.50','HAMBURGER',1);
```

```
INSERT INTO XXSTORE (ID, NAME, XXORDER_ID) VALUES (1,'CHICAGO',1);
```

```
INSERT INTO XXSALES (ID, NAME, XXSTORE_ID, XXORDER_ID) VALUES (1,'JOHN',1,1);
```








Welcome in SQL world



Let's do Java

Finally we are fully prepared to try some Java and SQL stuff.

Let's do Java

- ▼  XXORDER_TEST
 - ▼  src
 - ▼  xxorder
 - >  XXOrder.java
 - >  JRE System Library [JavaSE-1.8]
 - ▼  Referenced Libraries
 - >  imsudb.jar - C:\JDBC Drivers\IMS

```
package xxorder;  
  
import java.sql.Connection;  
import java.sql.SQLException;  
import java.sql.ResultSet;  
import java.sql.Statement;  
  
import com.ibm.ims.jdbc.IMSDataSource;
```

Let's do Java

JDBC DataSource Interface

JDBC DriverManager Interface

Let's do Java

- JDBC DataSource Interface

```
IMSDatasource ds = new IMSDataSource();  
ds.setDatabaseName("MYPSB");  
ds.setDatastoreName("IMS1");  
ds.setDatastoreServer("my.host.com");  
ds.setPortNumber(5555);  
ds.setDriverType(IMSDatasource.DRIVER_TYPE_4);  
ds.setUser("MyUserID");  
ds.setPassword("MyPassword");  
  
try {  
    Connection conn = ds.getConnection();  
} catch (SQLException e) {  
  
}
```

Let's do Java

- JDBC DriverManager Interface

```
try {  
    Class.forName("com.ibm.ims.jdbc.IMSDriver");  
} catch (ClassNotFoundException e1) {  
  
}  
  
String url = "jdbc:ims://my.host.com:5555/MYPSB";  
String user = "MyUserID";  
String password = "MyPassword";  
  
try {  
    Connection conn = DriverManager.getConnection(url, user, password);  
} catch (SQLException e) {  
  
}
```

Let's do Java

```
try {  
    Connection conn = ds.getConnection();  
    Statement st = conn.createStatement();  
    ResultSet rs = st.executeQuery("SELECT * FROM XXORDER");  
    while(rs.next()) {  
        System.out.println(rs.getString("ID") + " " +rs.getShort("ORDID"));  
    }  
  
    int insertRecords = st.executeUpdate("INSERT INTO XXORDER (ID, ORDID) VALUES (1, 18001)");  
    System.out.println("Inserted " + insertRecords + " Record(s)");  
  
    conn.commit();  
    conn.close();  
} catch (SQLException e) {  
  
}
```

Thank You.