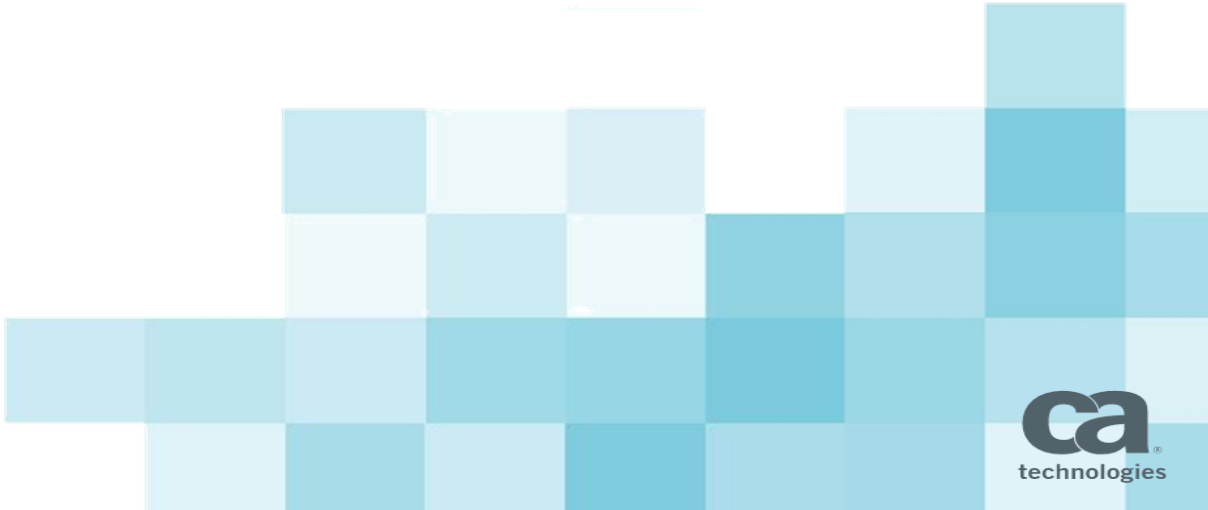
CA IDMS™ 19.0
SQL Enhancements for Modernization

Dave Ross

CA Technologies



EIUA Technical Conferences
September 2016

Abstract

Customers need to leverage their investment in CA IDMS to create new applications based on services in the application economy, and they need to do this with current technologies and developer skill sets. This session shows how SQL enhancements in CA IDMS 19.0 improve standards compliance, developer productivity, and compatibility with third party tools and applications. The SQL virtual foreign key feature enables developers to use standard SQL to access and update network databases without the need to use network DML or table procedures. Enhancements to SQL DDL enable users to define databases using standard DDL compatible with other databases.

© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

Agenda

1. SQL VIRTUAL FOREIGN KEYS
2. ISO STANDARD SQL CONSTRAINT DDL

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

CA IDMS SQL Relational to Network Mapping

SQL		Network	
Table	• Identically named	Record definition	• Via SCHEMA
Row	• Equivalent	Record occurrence	• Equivalent
Column	• Automatically renamed	Element	• Except ODO, redefines, …
Referential constraint	• Partial mapping	Set	• Lack of foreign keys

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Relational To Network Mapping Techniques

Syntax extensions

Views

Procedures

- Table procedures
- Called procedures

Foreign keys

- Referential sets

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Comparison of Mapping Techniques

	Standard SQL	New programs	Application changes	Restructure	Exposes Sets
SQL Extensions	No	No	No	No	Limited
Views	Yes	No	No	No	Limited
Table Procedures	Yes	Yes	No	No	Encapsulate
Referential Sets	Yes	No	Yes	Maybe	Referential constraints

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Virtual Foreign Keys

Expose Sets as Referential Constraints

- Queries
- Updates

Based on ROWID

- Virtual primary key = ROWID
- Virtual foreign key = FKEY_<set_name>

Define on SQL Schema definition

Visible through JDBC and ODBC metadata

Enable access for common tools and application frameworks

Use SQL to access and maintain network databases from Java

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Virtual Foreign Key Benefits

- Use standard SQL to access and update CA IDMS
  - Removes INSERT and UPDATE limitations
  - Capabilities like network DML
  - Write applications using Java, .NET, and popular frameworks
  - No special CA IDMS SQL syntax
- Easy to implement
  - No database changes
  - No changes to network definitions
  - No need for table procedures

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Virtual Primary Key

- Pseudo column ROWID
  - Uniquely identifies a row
  - Available for every table
  - Not nullable
- Datatype ROWID
  - Length 8 bytes
  - Contains DBKEY and Page Info
  - Value not persistent and thus not a true primary key
  - Relational operator support extended (<,>,<=,>=)

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Virtual Foreign Key

- Pseudo column FKEY_setname
  - “setname” is the SQL transformed name of set
  - Value is ROWID of owner of set-name or NULL if no membership
  - Available for every set in which record is member
  - Always appears nullable, but cannot set to NULL for MA sets
- Datatype ROWID
  - Must be 8 bytes
  - contains DBKEY and Page Info of set owner

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Virtual Foreign Keys

- Virtual foreign keys can be referenced in SQL statements like a user-defined column
- Included in list of “all” columns (SELECT *)
- Included in INSERT and UPDATE default column list
  - When column list is excluded
  - Column order:
    - User defined columns
    - ROWID
    - Virtual foreign keys in alphabetical order

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Enabling Virtual Foreign Keys

- "Create" virtual keys with SQL schema definition

```
create schema EMPSCHM
      for nonsql schema EMPDICT.EMPSCHM version 100
      with virtual keys
```

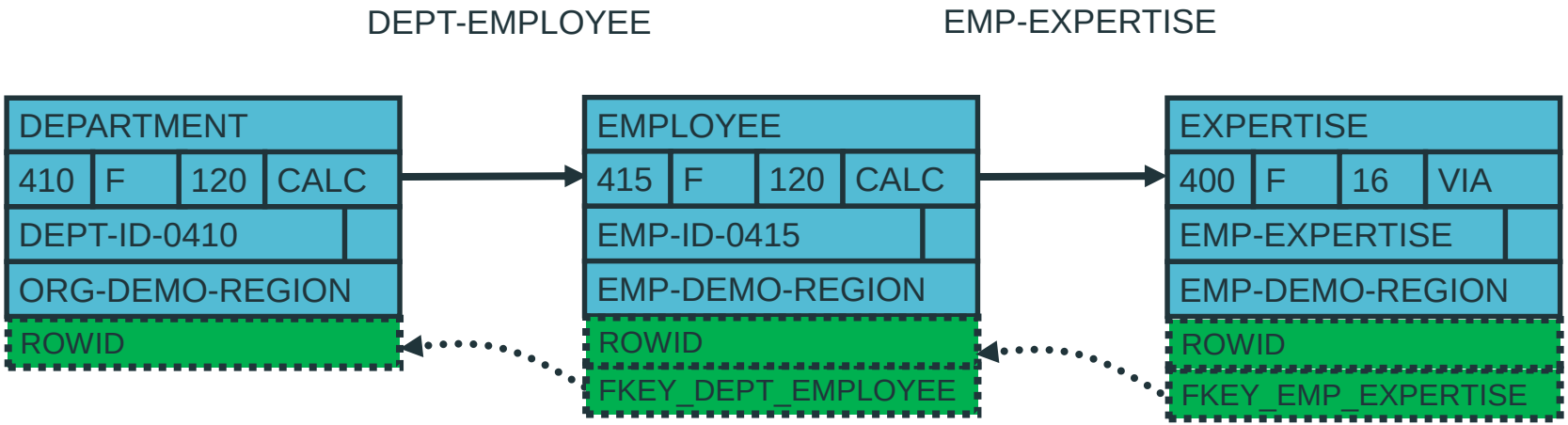
- ROWID and virtual foreign keys become visible
  - SELECT *
  - INSERT column list when list is omitted
  - UPDATE column list when list is omitted

- "Remove" virtual keys

```
alter schema EMPSCHM without virtual keys
```


© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

EMPSCHM With Virtual Foreign Keys



© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

SELECT Examples

- Obtain DEPARTMENT data of an EMPLOYEE

```
select *
  from DEPARTMENT where ROWID =
    (select FKEY_DEPT_EMPLOYEE
       from EMPLOYEE
      where EMP_ID = 1001)
```

- Join DEPARTMENT and EMPLOYEE

```
select d.*, e.*
  from DEPARTMENT d, EMPLOYEE e
     where d.ROWID = e.FKEY_DEPT_EMPLOYEE
```


© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

More SELECT Examples

- Select all EXPERTISE records for an EMPLOYEE

```
select * from EXPERTISE
  where FKEY_EMP_EXPERTISE =
      (select ROWID
        from EMPLOYEE
        where EMP_ID = 1001)
```

- Show all DEPARTMENTs including those without EMPLOYEEs

```
select d.*, e.*
  from DEPARTMENT d left join EMPLOYEE e
    on d.ROWID = e.FKEY_DEPT_EMPLOYEE
```


© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

INSERT using Virtual Keys

- Use to establish set membership
  - Set virtual foreign key column to owner ROWID or NULL
  - Set membership options define integrity constraints
  - MA set disallows NULL valued virtual foreign key
- Use ROWID to suggest DBKEY for a DIRECT record
  - Defaults to -1 on STORE (first available DBKEY in page range used)
  - Ignored if record location mode is not DIRECT

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

INSERT using Virtual Keys

- Use of scalar-query-expression in VALUES clause
  - New R19.0 feature for INSERT statements for all SQL schemas
  - With virtual keys, allows INSERT to contain queries that retrieve the virtual foreign key values to be set in the inserted row
- Support for virtual keys in query-specification
  - Alternative to VALUES clause on INSERT
  - Query result must have same number of columns as named in INSERT statement
  - If no columns named, number of columns must match the total number of columns, including ROWID and all VFKs

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

INSERT Examples

VALUES clause with column list

```
insert into EMPLOYEE
        (EMP_ID,
        EMP_FNAME,
        EMP_LNAME,
        FKEY_DEPT_EMPLOYEE,
        FKEY_OFFICE_EMPLOYEE)
 values
        (976,
        ’SEBASTIAN’,
        ’VOLLMER’,
        (select ROWID from DEPARTMENT where DEPT_ID = 9003),
        (select ROWID from OFFICE where OFFICE_CODE = '002'))
```


© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

INSERT Examples

VALUES clause without column list

```
insert into EMPLOYEE
 values
        (976,
        'SEBASTIAN',
        'VOLLMER',
        NULL,
        (select ROWID from DEPARTMENT where DEPT_ID = 9003),
        (select ROWID from OFFICE where OFFICE_CODE = '002'))
```

- EMPLOYEE columns are
  EMP_ID, EMP_FNAME, EMP_LNAME, ROWID,
  FKEY_DEPT_EMPLOYEE, FKEY_OFFICE_EMPLOYEE

© 2016 CA. ALL RIGHTS RESERVED.

INSERT Examples

Specifying virtual keys using query-specification

```
insert into EMPOSITION
        (START_YEAR,
        FKEY_EMP_EMPOSITION,
        FKEY_JOB_EMPOSITION)
 select 2015, E.ROWID, J.ROWID from EMPLOYEE E, JOB J
        where E.EMP_ID = 23 and J.JOB_ID = 2025
```


© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

UPDATE

- Virtual Foreign Keys can be specified in SET clause
- Equivalent to CONNECT/DISCONNECT DML verbs
- Allowed for sets with membership options
  - OPTIONAL (OA/OM)
  - MANDATORY MANUAL (MM) when record not yet connected to set
- SET FKEY_setname values allowed
  - NULL
  - ROWID of owner occurrence of setname
  - ROWID of member occurrence of setname

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

UPDATE Examples

- Disconnecting the EMPLOYEE from the DEPT-EMPL set

```
update EMPLOYEE set FKEY_DEPT_EMPL = NULL
        where EMP_ID = 23
```

- EMPLOYEE connected to DEPARTMENT 4000

```
update EMPLOYEE set FKEY_DEPT_EMPL =
                (select ROWID from DEPARTMENT where DEPT_ID=4000)
        where EMP_ID = 23
```


© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

DELETE

- Use virtual key columns the same as user-defined columns

```
delete from EMPLOYEE
  where FKEY_DEPT_EMPL =
        (select ROWID from DEPARTMENT
              where DEPT_NAME ='Sales')
```

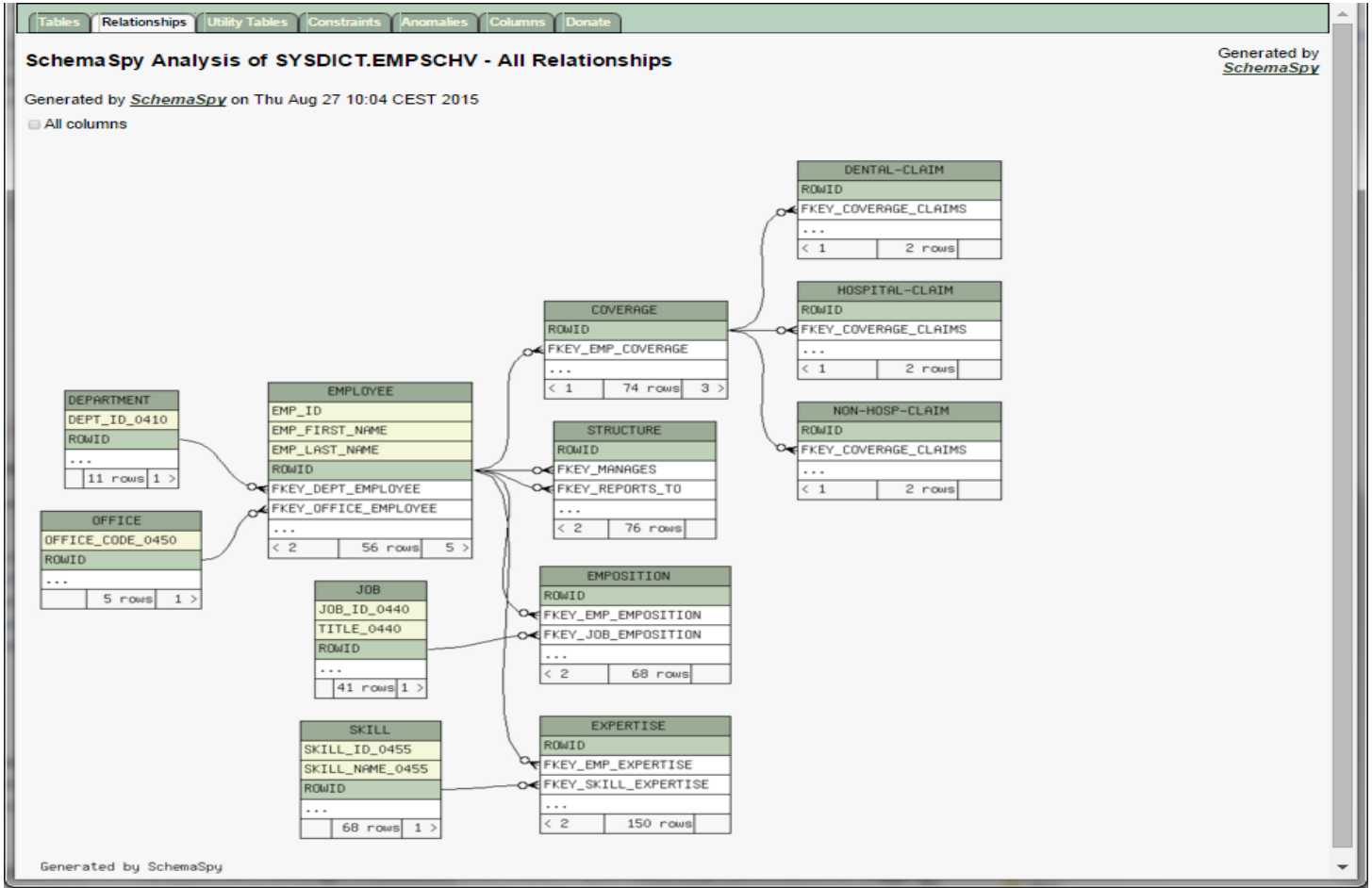

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Discovering Virtual Keys

- JDBC and ODBC metadata APIs expose virtual keys
- java.sql.DatabaseMetaData interface methods
  - getPrimaryKeys
  - getExportedKeys
  - getImportedKeys
  - getCrossReference
- ODBC API functions
  - SQLPrimaryKeys
  - SQLForeignKeys

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Discovering Network Relationships



ca technologies

Virtual Keys Compared to other Mapping Techniques

	Standard SQL	New programs	Application changes	Restructure	Exposes Sets
SQL Extensions	**No**	**No**	**No**	**No**	**Limited**
Views	**Yes**	**No**	**No**	**No**	**Limited**
Table Procedures	**Yes**	**Yes**	**No**	**No**	**Encapsulate**
Referential Sets	**Yes**	**No**	**Yes**	**Maybe**	**Referential constraints**
Virtual Keys	**Yes**	**No**	**No**	**No**	**Referential constraints**

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

SQL DDL Enhancements

- ISO Standard Constraint Definition DDL
  - CREATE TABLE
  - ALTER TABLE
  - Unique constraints
  - Primary Key constraints
  - Referential constraints
- Improves integration with frameworks and tools
- DISPLAY TABLE

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Defining UNIQUE Constraints

- A unique constraint is specified by coding
  - UNIQUE on a column definition
  - UNIQUE (<column-names>) in the column list
- A constraint may be named
  - CONSTRAINT <name>
  - Precedes the constraint definition
  - Name is generated if not named
- More than one unique constraint may be defined

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

UNIQUE Constraint Example

```
create table ABC.TABLE1(
        COL1 CHAR(8) unique,
        COL2 CHAR(8),
        constraint CON1 unique (COL2, COL1))
```


© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

Defining PRIMARY KEY Constraints

- A Primary Key constraint is specified by coding
  - PRIMARY KEY on a column definition
  - PRIMARY KEY (<column-names>) in the column list
- A constraint may be named
  - CONSTRAINT <name>
  - Precedes the constraint definition
  - Name is generated if not named
- ONE Primary Key constraint per table

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Primary Key Constraint Examples

```
create table ABC.TABLEX (
        COL1 char(8) not null primary key,
        COL2 char(8) not null,
        COL3 char(8) not null)

create table ABC.TABLEY (
        COL1 char(8) not null,
        COL2 char(8) not null,
        COL3 char(8) not null,
        primary key (COL2, COL3))
```


© 2016 CA. ALL RIGHTS RESERVED.

ca technologies

Generated Enforcing Index

- Enforces unique and primary key constraints
- Created by CREATE TABLE and ALTER TABLE
  - Suppresses creation of default index
  - Uses default values for index attributes
  - Cannot be altered to be NOT UNIQUE
  - Cannot be dropped with DROP INDEX unless replacement defined
- Generated Index names
  - IDX################
  - ### is a 15 digit number unique in the schema

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Replacing a Generated Enforcing Index

- Use CREATE UNIQUE INDEX
- Columns must match enforcing index
- Inherits enforcing index attribute
- When primary key index dropped
  - A replacement inherits the primary key attribute
- Generated indexes dropped automatically
- Non-generated indexes must be manually dropped

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Defining Referential Constraints

- A referential constraint is specified by coding
  - REFERENCES clause on a column definition
  - FOREIGN KEY clause in the column list
- A constraint may be named
  - CONSTRAINT name
  - Precedes the constraint definition
  - If not named, a name is generated
    - GEN################
- More than one referential constraint may be defined

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Referential Constraints Example

```
create table ABC.TABLEX (
        COL1 char (8),
        COL2 char(8),
        foreign key (COL1,COL2) references TABLEP,
        COL3 char(8) not null
              constraint CON1  references TABLEP (COL5))
```


© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Discovering Constraints

- java.sql.DatabaseMetaData interface methods
  - getIndexInfo
  - getPrimaryKeys
  - getExportedKeys
  - getImportedKeys
  - getCrossReference
- ODBC API functions
  - SQLStatistics
  - SQLPrimaryKeys
  - SQLForeignKeys

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

Summary

- CA IDMS 19.0 enables Java and Windows developers to use industry standard technology to create, access, and maintain CA IDMS databases
- SQL Virtual Key feature enables developers to use standard SQL to access and update network databases
- SQL DDL enhancements enable CA IDMS to support database definitions provided by popular tools and frameworks

© 2016 CA. ALL RIGHTS RESERVED.
ca technologies

FOR INFORMATION PURPOSES ONLY
Terms of this Presentation

This presentation was based on current information and resource allocations as of May 2016 and is subject to change or withdrawal by CA at any time without notice. Notwithstanding anything in this presentation to the contrary, this presentation shall not serve to (i) affect the rights and/or obligations of CA or its licensees under any existing or future written license agreement or services agreement relating to any CA software product; or (ii) amend any product documentation or specifications for any CA software product. 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 will make such release available (i) for sale to new licensees of such product; and (ii) to existing licensees of such product on a when and if-available basis as part of CA maintenance and support, and in the form of a regularly scheduled major product release. Such releases may be made available to current licensees of such product who are current subscribers to CA maintenance and support on a when and if-available basis. In the event of a conflict between the terms of this paragraph and any other information contained in this presentation, the terms of this paragraph shall govern.

Certain information in this presentation may outline CA's general product direction. All information in this presentation is for your informational purposes only and may not be incorporated into any contract. CA assumes no responsibility for the accuracy or completeness of the information. To the extent permitted by applicable law, CA provides this presentation "as is" without warranty of any kind, including without limitation, any implied warranties or merchantability, fitness for a particular purpose, or non-infringement. In no event will CA be liable for any loss or damage, direct or indirect, from the use of this document, 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. CA confidential and proprietary. No unauthorized copying or distribution permitted.

© 2016 CA. ALL RIGHTS RESERVED.

David Ross
Sr. Principal Product Owner
David.Ross@ca.com