Advanced DataTools Webcast

from the IBM Informix Champions

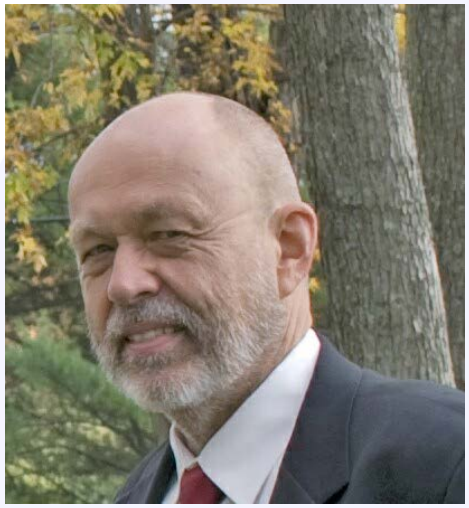
Informix Tutorial Databases, Tables, System Catalogs and Managing Security

by Lester Knutsen

Thursday, July 23, 2020 2:00pm EDT

Advanced DataTools

Lester Knutsen

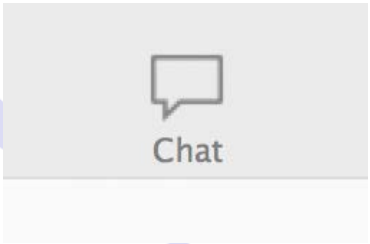


Lester Knutsen is President of Advanced DataTools Corporation and has been building large data warehouse and business systems using Informix Database software since 1983. Lester focuses on large database performance tuning, training, and consulting. Lester is a member of the IBM Gold Consultant program and was presented with one of the Inaugural IBM Information Champion awards by IBM. Lester was one of the founders of the International Informix Users Group and the Washington Area Informix User Group.

lester@advancedatatools.com
www.advancedatatools.com
703-256-0267 x102

Webcast Guidelines

- The Webcast is pre-recorded. The Webcast replay and slides will be available after the broadcast.
- Please Mute your line - background sounds will distract everyone.
- Use the Chat Button in the upper right to ask questions.



Agenda

- Creating Databases
- Database Logging Modes
- Creating Tables
- Exploring the Database Catalogs
- Managing Database Security

Examples and Demos

- Object Oriented Database Tables
- Using SQL on the System Catalogs to Create SQL Scripts
- Creating Non-Operating System Users for Informix Databases

Creating Databases

Create Database

- **Database Location**
  - The dbspace that will store the system catalogs and default for all tables.
  - Default is rootdbs.
  - A database should NOT be in the rootdbs.
- **Database Logging Mode**
  - One of the four logging modes for Informix databases.
  - Default is No Logging.
  - Recommend production databases be created with unbuffered logging.
  - Read-only databases or databases where transactions are not required can be created with no logging.

Syntax

```
CREATE DATABASE database_name
   [ IN dbspace_name ]
   [ WITH  | LOG
           | BUFFERED LOG
           | LOG MODE ANSI ]
```

Database Logging Modes

- No Logging
- Buffered Logging
- Unbuffered Logging
- ANSI Mode Logging

No Logging

- Fastest but least safe
- Transactions are not used
- Commit or Rollback work statements not allowed

Data recovery:

- From the last checkpoint if physical log disk is available
- From last archive is physical log disk is not available

- Faster inserts, updates, and deletes
- Note: Logical Logs are used for some internal activities

UnBuffered Logging

- Slowest and safest
- Transactions can be used

Data recovery:

- From last committed transaction if logical log disk is OK
- From last committed transaction on logical log tape

- All transactions are immediately written to disk (slowest performance)
- Need to monitor logs so they do not fill up

Buffered Logging

- Better performance but not as safe
- Transactions may be used

Data recovery:

- From last committed transaction flushed to disk
- From last committed transaction flushed to logical log tape

- Logs stored in memory buffer until written to disk
- Need to monitor logs so they do not fill up

ANSI Mode Logging

- Enforces ANSI rules for transaction processing
- Always in a transaction (No begin work)
- Similar to Unbuffered Logging

Creating Tables

Create Table

- **Table Name**
  - Every table should have a unique name within the database. Recommend that temporary table names begin with the word "tmp_" to identify that they are temporary.
- **Columns**
  - The columns identify the data elements that make up a table schema.
- **Primary Key**
  - Every table should have a Primary Key. This is one or more columns that define a unique identifier for a table.
- **Storage Requirements**
  - Projected size of the table. The storage requirements are used in the EXTENT OPTION of the Create Table statement and to determine which DBSPACE the table will be created in.
- **Lock Mode**
  - Specific Row- or Page-level locking for a table. Default is Page.
- **Other Constraints**
  - Include Default Values, Foreign Keys, Check Conditions, and Constraint Names. These are described in the Informix Guide to SQL: Reference Manual.

Create Table Syntax

```
CREATE [TEMP] TABLE table_name
    ( column_name       datatype, [ column constraint ]
    column_name         datatype...
    [ table constraint ]
)
    [ WITH NO LOG ]
    [ IN dbspace ]
    [ FRAGMENT BY clause ]
    [ EXTENT SIZE x ]
    [ NEXTSIZE x ]
    [ LOCK MODE PAGE | ROW ];
```

Character Data Types

- CHAR - Stores any string of letters, numbers, and symbols
- VARCHAR - Stores character strings of varying length
- NCHAR - Stores any native string of letters, numbers, and symbols
- NVARCHAR - Stores native character strings of varying length
- LVARCHAR - Stores variable length character data type (32k maximum)

Numeric Data Types

- INTEGER - Stores whole numbers from -2,147,483,647 to +2,147,483,647
- **INT8 - (8 bytes) Range is -9,223,372,036,854,775,807 to +9,223,372,036,854,775,807**
- SMALLINT - Stores whole numbers from -32,767 to +32,767
- SERIAL - Stores sequential integers
- SERIAL8 - (8 bytes)
- SMALLFLOAT - Stores single-precision floating numbers corresponding to the float data type in C
- FLOAT - Stores double-precision floating numbers corresponding to the double data type in C
- DECIMAL - Stores numbers with definable scale and precision
- MONEY - Stores currency amount

Date and Time Data Types

- DATE - Stores calendar date
- DATETIME - Stores calendar date combined with time of day
- INTERVAL - Stores span of time

Other Data Types

- TEXT - Stores any kind of text data
- BYTE - Stores any kind of binary data

- BOOLEAN - valid values: “t”, “f”, null; case sensitive

JSON and BSON

- JSON and BSON data types are used to contain JSON or BSON data
- JSON or BSON columns contain multiple documents, each similar to a row in a relational database. Documents can be nested.
- The JSON data type contains plain text.
- The BSON data type is a binary format of the JSON data type.
- BSON data type is better suited for storing structured data in an Informix database. The database server can operate on data in BSON columns but can only insert and display data in JSON columns.
- JSON and BSON documents up to 4 KB are stored in-row. Documents greater than 4 KB in size are stored in the sbspace that is associated with the table, or the default subspace.
- The maximum size of a JSON or BSON document is 32 KB. The maximum size of a JSON or BSON column is limited only by the operating system.

Object Oriented Data Types and Extensibility

Object Oriented Data Types

Mechanism	Example	Strengths and Weaknesses
Built-In Types	INTEGER, VARCHAR, DATE etc. These are standardized in the SQL-92 language specification.	Mature and high performance because they are compiled into the ORDBMS. But they are very simple. Good building-blocks for other types.
DISTINCT	CREATE DISTINCT TYPE String AS VARCHAR(32);	Simple to create, and useful when what you want is something very close to another type.
ROW TYPES	CREATE ROW TYPE Address (Address_Line_One String NOT NULL, Address_Line_Two String NOT NULL, City String NOT NULL, State String, ZipCode PostCode, Country String NOT NULL);	Relatively easy to use means of combining pre-existing types into a more complex objects, and enforcing rules about contents. ROW TYPEs have several drawbacks that makes them a poor choice for types to define columns.
Java Classes	Combination of Java UDRs with opaque data storage.	More complex to develop, but an excellent choice when you want code that runs in both the outside, and inside, the DBMS.
OPAQUE TYPES	CREATE OPAQUE TYPE GeoPoint (internallength = 16);	Most complex to develop, but these are the most powerful in terms of performance, scalability and the range of object sizes that can be supported.

Row Data Types Example:

```
CREATE TABLE Employees (
    Id                  Employee_Number          PRIMARY KEY,
    Name                PersonName               NOT NULL,
    Date_of_Birth       DATE                     NOT NULL,
    Address             MailAddress              NOT NULL,
    LivesAt             st_point                 NOT NULL,
    Resume              Document                 NOT NULL,
    Voice_Key           Audio_Recording          NOT NULL,
    Holidays            SET ( Period NOT NULL )              );
```

Row Data Types

- Analogous to C structure, come in two “kinds”
  - NAMED - strongly typed, ID’d by name, has inheritance, used to build columns and tables
  - UNNAMED - weakly typed, ID’d by structure, no inheritance, used to build columns
- Can contain built-in, collection, opaque, distinct, and other row type data types
  - Caveat: no serial or serial8

Row Data Types

- Advantages
  - Less coding
  - Refers to a group of elements by a single name
  - Intuitive
- Disadvantages
  - More complex
  - Not simple SQL
  - Sys Adm is more complex
  - No “alter type” statement, must drop and recreate

Example Row Data Type - Named

Row Data Types Named:

```
create row type name_t
   (fname char(20), lname char(20));

create row type address_t
   (street_1 char(20), street_2 char(20),
   city char(20), state char(2), zip char(9));

create table student
   (student_id serial,
   name         name_t,
   address      address_t,
   company      char(30));
```

Using Named Row Types in SQL

Insert statement:

```
insert into student
values (1234,
   row (“John”,”Doe”)::name_t,
   row ("1234 Main Street","",
   "Anytown","TX","75022")::address_t,
   "Informix Software")
```

Select statement:

```
select * from student where name_t.lname matches "Doe”;
Result set:
student_id  1234
name        ROW('John ','Doe')
address     ROW('1234 Main Street',’,'Anytown','TX','75022')
company     Informix Software
```

Collections

- Grouping of elements of the same datatype (char, int), max size = 32 KB
- Used when
  - The data is meaningless without the context of the other members in the collection (e.g., golf scores, *to do* list, set of names)
  - Individual data elements are not likely to be directly queried by position
  - The maximum number of data elements is less than 32
- Can be null

Collections

Three kinds of collections:

- Set - unordered, no duplicates allowed set
  - {“apple”, ”orange”, ”grapefruit”, “plum”}
- Multiset - unordered, duplicates allowed multiset
  - {“apple”, “orange”, “grapefruit”, “apple”, “plum”, “grapefruit”}
- List - ordered, duplicates allowed list
  - {“apple”, “orange”, “grapefruit”, “apple”, “plum”, “grapefruit”}

Collections - Example

create table class

(class_id serial,

class_name varchar(60),

description lvarchar,

prereqs set(char(20) not null));

Insert syntax is similar to named row types:

insert into class values (300,

“Advanced Informix Performance Tuning”,

“Covers advanced information on tuning the Informix Database Server”,

(SET{“InformixDBA”,”InformixSQL”,”Unix”}));

User-Defined Data Types - Distinct

- Two user-defined data types (UDTs):
  - Distinct
    - data type modeled on an existing data type
    - has a unique name to distinguish it from other similar “types”
    - inherits the internal structure from the source type
    - might have operations and casts defined over its source type
  - Opaque
    - data type that is unknown to the database server
    - you must define the internal structure, functions, and operations

Partitioning Tables

Partitioning Tables

- See Webcast: Automatic Informix Range Interval Partitioning and Rolling Windows to Organize your data on June 21, 2018

https://advancedatatools.com/webcasts/informix-webcast-2018-automatic-informix-range-interval-partitioning-and-rolling-windows-to-organize-your-data/

Exploring the Database Catalogs

Exploring the Database Catalogs

- Use System Catalogs to get information about your database and tables
- Use System Catalogs to Generate SQL to perform database administration tasks

Systables

Select * from systables where tabid > 99

```
tabname         state
owner           informix
partnum         3145868
tabid           100
rowsize         26
ncols           3
nindexes        1
nrows           52.00000000000
created         06/30/2020
version         6553622
tabtype         T
locklevel       P
npused          1.000000000000
fextsize        16
nextsize        16
flags           0
site
dbname
type_xid        0
am_id           0
pagesize        2048
ustlowts        2020-07-04 01:11:02.00000
secpolicyid     0
protgranularity
statchange
statlevel       A
```

Systables

Column	Value	Definitin
tabname	state	Name of Table
owner	informix	Table Creator
partnum	3145868	Unique address of Table on Server
tabid	100	Table ID in Database
rowsize	26	Size of Row in Bytes
ncols	3	Number of Columns
nindexes	1	Number of Indexes
nrows	52	Number of Rows (after last Update Statistics)
created	6/30/20	Data Table Created
version	6553622	Vesion
tabtype	T	Type of Table: T = Table E = External Table V = View Q = Sequence P = Private synonym S = Public synonym
locklevel	P	Lock mode for the table: B = Page and row level P = Page level R = Row level
npused	1	Number of pages used
fextsize	16	First Extent Size
nextsize	16	Next Extent Dize
flags	0	Flags to Identify: ROWID - 1 - Has rowid column defined UNDER - 2 - Table created under a supertable VIEWREMOTE - 4 - View is based on a remote table CDR - 8 - Has CDRCOLS defined RAW - 16 - (Informix) RAW table EXTERNAL - 32- External table AUDIT - 64 - Audit table attribute - FGA AQT - 128 - View is an AQT for DWA offloading VIRTAQT - 256 - View is a virtual AQT
site		Future
dbname		Future
type_xid	0	Internal
am_id	0	Internal
pagesize	2048	Table Page Size
ustlowts	7/4/20 1:11 AM	Last Low Update Stats
secpolicyid	0	Security ID when used
protgranularity		LBAC Level when used
statchange		Internal
statlevel	A	Level of Update Statistics: T = table F = fragment A = automatic

Exploring the Database Catalogs

Informix 14.10.FC3 Database Catalog Table ID and Names

1	systables	21	sysopclstr	41	systabamdata	61	sysexternal
2	syscolumns	22	systriggers	42	sysopclasses	62	sysextdfiles
3	sysindices	23	systrigbody	43	syserrors	63	sysextcols
4	systabauth	24	sysdistrib	44	systraceclasses	64	sysautolocate
5	syscolauth	25	sysfragments	45	systracemsgs	65	sysfragdist
6	sysviews	26	sysobjstate	46	sysaggregates	70	sysdomains
7	sysusers	27	sysviolations	47	syssequences	71	sysindexes
8	sysdepend	28	sysfragauth	48	sysdirectives	90	GL_COLLATE
9	syssynonyms	29	sysroleauth	49	sysxasourcetypes	91	GL_CTYPE
10	syssyntable	30	sysxtdtypes	50	sysxadatasources	99	VERSION
11	sysconstraints	31	sysattrtypes	51	sysseclabelcomponents		
12	sysreferences	32	sysxtddesc	52	sysseclabelcomponentelement		
13	syschecks	33	sysinherits	53	syssecpolicies		
14	sysdefaults	34	syscolattribs	54	syssecpolicycomponents		
15	syscoldepend	35	syslogmap	55	syssecpolicyexemptions		
16	sysprocedures	36	syscasts	56	sysseclabels		
17	sysprocbody	37	sysxtdtypeauth	57	sysseclabelnames		
18	sysprocplan	38	sysroutinelangs	58	sysseclabelauth		
19	sysprocauth	39	syslangauth	59	syssurrogateauth		
20	sysblobs	40	sysams	60	sysproccolumns		

Using the System Tables to Generate SQL

- Write the SQL one time
- Use to SQL on any Database or Server to accomplish the task

mkSQL_revoke_public.sql

```
-- Output Section
output to "o_revoke_public.sql"
without headings
-- Syntax revoke all on tablename from public;
select "revoke all on " ,
        trim(tabname) ,
        " from public;"
from    systables , systabauth
where   systables.tabid = systabauth.tabid
and grantee = "public"          -- tables that have public access
and systables.tabid > 99;       -- non system tables
```

Structure of SQL script to generate SQL

1– Basic Select Statement

```
select
        tabname
from    systables , systabauth
where   systables.tabid = systabauth.tabid
and     grantee = "public" -- tables that have public access
and systables.tabid > 99; -- non system tables
```

Structure of SQL script to generate SQL

2 – Add the Correct Syntax to generate:

revoke all on tablename from public;

```
select "revoke all on " ,
        trim(tabname) ,
        " from public;"
from    systables , systabauth
where   systables.tabid = systabauth.tabid
and grantee = "public"          -- tables that have public access
and systables.tabid > 99;       -- non system tables
```

Structure of SQL script to generate SQL

3 – Add the Output Section

-- Output Section

output to "o_revoke_public.sql” without headings - No headings and column names

Other Output Options

- Unload statement
- Insert into external table

Dbaccess Limitation

- Dbaccess is limited to 80 character output
  - Lines greater than 80 characters will wrap in the wrong place – may lead to syntax errors
  - Best practice is to separate lines and variables that will expand
  - Or use Unload Statement and set a Delimiter

mkSQL_disable_PK.sql

This example may wrap at 80 characters and lead to Syntax errors

```
output to "o_disable_PK.sql"
without headings
select "set constraints " ll trim(constrname) ll " disabled ;"
from sysconstraints
where tabid > 99        -- non system tables
and constrtype = "P";   -- Constraint type is Primary Key
```

mkSQL_disable_PK.sql

Best practice is to keep syntax text and variables on separate lines

```
output to "o_disable_PK.sql"
without headings
select     "set constraints ”,
           trim(constrname),
           " disabled ;"
from sysconstraints
where tabid > 99        -- non system tables
and constrtype = "P";   -- Constraint type is Primary Key
```

Data Source – System Tables

- Systables
- Syscolumns
- Syscnstraints
- Sysusers
- Systabauth
- Sysindexes
- Sysprocedures

Examples – More Scripts to Generate SQL

- mkSQL_alter_row_lock.sql
- mkSQL_create_PK.sql
- mkSQL_dbspace_script.sql
- mkSQL_drop_constraint.sql
- mkSQL_llogs_script.sql
- mkSQL_PK_disable.sql
- mkSQL_revoke_public.sql
- mkSQL_update_stats_low.sql
- mkSQL_update_stats_med.sql

Alter Table to Row Level Locking

```
output to o_alter_row_lock.sql without headings
select    "alter table ",
          trim(tabname) ,
          "lock mode ( row );"
from systables
where locklevel = "P"   -- tables with page level locking
and tabtype = "T"       -- real tables
and tabid > 99          -- non system tables
```

Enable Primary Keys

```
output to "o_enable_PK.sql"
without headings
select    "set constraints ",
          trim(constrname),
          "enabled;"
from sysconstraints
where tabid > 99        -- non system tables
and constrtype = "P"
order by constrid desc; -- Constraint type is Primary Key
```

Drop Constraints

```
output to "o_drop_constraints.sql"
without headings
select  "alter table ",
        trim(tabname) ,
        "drop constraint ",
        trim(sysconstraints.constrname),
        ";"
from sysconstraints, systables
where sysconstraints.tabid      = systables.tabid
and sysconstraints.constrtype = "R" ;  - Reference Constraints
```

Alter Table to Add Primary Keys

```
--- Script to alter tables with unique index to add a primary key
--- Get tables with no primary key and a unique index
output to "o_creat_pk.sql"
without headings
select "alter table ",
        trim(tabname) ,
        " add constraint primary key (",
        trim(colname), ") ;"
from systables, sysindexes, syscolumns
where systables.tabid = sysindexes.tabid
and   systables.tabid = syscolumns.tabid
and systables.tabid > 99
and systables.tabid not in ( select tabid from sysconstraints where constrtype = "P" )
and sysindexes.idxtype = "U"
and sysindexes.part1 = syscolumns.colno;
```

Update Statistics Low

```
output to o_update_stats_low.sql
without headings
select "update statistics low for table ",
        trim (tabname) ll ";"
from systables where tabid > 99
and tabtype = "T";
```

Update Statistics Medium

```
output to o_update_stats_med.sql
without headings
select "update statistics medium for table ",
        trim (tabname) ,
        "distributions only;"
from systables where tabid > 99
and tabtype = "T";
```

Update Statistics High

```
output to o_update_stats_high.sql
without headings
select "update statistics high for table ",
        trim (tabname) ,
        " ( ",
        trim (colname) ,
        " ) ",
        "distributions only;"
from sysindexes i, systables t, syscolumns c
where i.tabid = t.tabid
and i.tabid = c.tabid
and i.part1 = c.colno
and i.tabid > 99
and tabtype = "T";
```

Drop Distributions

```
output to o_update_stats_drop_dist.sql
without headings
select "update statistics low for table ",
        trim (tabname) ,
        "drop distributions;"
from systables where tabid > 99
and tabtype = "T";
```

Managing Database Security

Managing Database Security

- Database Level Permissions
- Table Level Permissions
- Column Level Permissions
- Functions and Stored Procedures Permissions

Database Level Privileges

- **DBA** - DBA privilege allows connect and resource privileges, and also gives the user the authority to grant resource, connect or DBA to another user. A user with DBA privilege can also drop any object in the database, and/or drop the database entirely.
  - DBA Database Owner - the user who originally created the database is automatically granted DBA
- **Resource** - Resource privilege allows a user to create new tables, indexes, and procedures in the database. The user can add, modify, and delete data from tables for which they have privileges.
- **Connect** - Connect privilege allows a user to access the database. The user can add, modify, and delete data from tables for which they have privileges.

Database Syntax

- grant [dba | resource | connect] to [public | USERS];
- revoke [dba | resource | connect] from [public | USERS];

Sysusers Table

- Database Privileges are in the System table: sysusers

Column name	Type
username	char(32)
usertype	char(1)
priority	smallint
password	char(16)
defrole	char(32)

Database Privileges Recommendations

- Avoid Public DBA Privileges.
- If you grant DBA to public any user can:
  - drop the database
  - revoke all other privileges from all other users
  - create new users of the database
- If you grant Resource to public - any user can create additional objects in the database (tables, views)
- Revoking DBA or Resource from a user converts their database privileges to CONNECT. You then need to revoke CONNECT to completely remove the user from the database.

Database Privileges Recommendations

Recommended database privileges:

- **DBA** - one or two logins that are not used by regular users. These logins should only be used for Database Administration functions. (DBA and DSO)
- **Resource** - only for programmers in a development environment or users who run processes that must create indexes.
- **Connect** - only users who need to access the database

Table Level Privileges

- **Select** - view data in a table
- **Insert** - add new rows of data to a table
- **Update** - change existing rows of data in a table
- **Delete** - remove rows of data from a table
- **Index** - add indexes to a table
- **Alter** - modify the table structure of a table
- **Reference**- ability to create referential constraints
  - A user must have Resource database privileges to index, alter, or create referential constraints on a permanent table.

Table Privileges Syntax

grant

```
[ all | [ select | insert | update | delete | index | alter | reference ]
on [ TABLE | VIEW | SYNONYM ]
to  [ public | USERS ]
[ with grant option ]
[ as GRANTOR ];
```

revoke

```
[ all | [ select | insert | update | delete | index | alter | reference ]
on [ TABLE | VIEW | SYNONYM ]
From [ public | USERS ];
```

Systabauth Table

- **Table Privileges are in the System table: systabauth**

Column name	Type
grantor	char(32)
grantee	char(32)
tabid	integer
tabauth	char(9)

Show ALL Users and Table Privileges

```
select  systables.tabname, systabauth.*
from    systables, systabauth
where systables.tabid =  systabauth.tabid
and     systables.tabid > 99
order by tabname;
```

Example

```
tabname  bills
grantor  informix
grantee  public
tabid    106
tabauth  su-idx---

tabname  customer
grantor  informix
grantee  public
tabid    101
tabauth  su-idx---

tabname  product
grantor  informix
grantee  public
tabid    102
tabauth  su-idx---
```

Informix Info Command

- info privileges for tabname;

```
info privileges for bills
```

User	Select	Update	Insert	Delete	Index	Alter
public	All	All	Yes	Yes	Yes	No

Table Recommendations

- Use Public privileges with care. You can revoke all table privileges from a user, BUT if public has any privileges, that user will have privileges.
- Recommended table privileges: In a production database, limit privileges to select, update, insert, and delete based on your requirements.
- Only the user who granted a privilege can revoke a privilege.
- The creator of a table is the owner and has full privileges. These privileges cannot be revoked.
- The "with grant" option to allows user to grant privileges.
- Use the "as GRANTOR" option to grant privileges *as* another user. Only a DBA or a USER with grant permissions can use this option.

Column Level Privileges

- **Select** - view specific columns in a table
- **Update** - change specific columns in a table
- **Reference** - ability to create referential constraints.

Column Privileges Syntax

```
grant [ select ( COLUMN ) |
        update ( COLUMN ) |
        reference ( COLUMN ) ]
on    [ TABLE | VIEW | SYNONYM ]
to    [ public | USERS ]
      [ with grant option ]
      [ as GRANTOR ] ;
```

Syscolauth Table

- Column privileges are in the System table: syscolauth

Column name	Type
grantor	char(32)
grantee	char(32)
tabid	integer
colno	smallint
colauth	char(3)

Show ALL Users with Column Privileges

```
select      tabname, colname, syscolauth.*
from        systables, syscolumns, syscolauth
where       systables.tabid = syscolumns.tabid
and         syscolumns.tabid = syscolauth.tabid
and         syscolumns.colno =
            syscolauth.colno
order by grantee;
```

Column Privileges Recommendations

- Column level privileges do not work unless you revoke select and update privileges at the table level.
- An asterisk will display in the third position of the column tabauth in the systabauth table when there are column level privileges.

tabname	grantor	grantee	tabid	tabauth
customer	lester	robert	100	su*id---

SPL, Procedure, Function, and Routine Privileges

- Execute on
  - Procedure
  - Function
  - Routine
- Example:
  - grant execute on procedure add_user_spl to ifxuser;
  - revoke execute on procedure add_user_spl from ifxuser;

Other Security Features

- Informix supports Label-based Access Controls (LBAC)
- Role Based Security
  - Assign Users to Roles
  - Assign Privileges to Roles
  - Set Default Roles

Owner Privileges

- The owner/creator of an object (database, table, etc.) has special privileges for that object.
  - Database - automatically has DBA privileges.
  - Table - automatically has all table level privileges.
- Owner privileges cannot be revoked from the user who created an object. This can create problems when a system moves from development to production or when the original owner of a table or database leaves the organization.
- Owner privileges are not easily identified. The "info" does not list owner's privileges.
- Another user, even a DBA, cannot revoke privileges from an owner.

Informix Defaults

- When an INFORMIX database is created, the database level privilege DBA is automatically granted to the creator.
- When a table is created, by default public is granted select, update, insert, and delete privileges on that table.
- When a view is created, no public privileges are granted.

Set Up Your Environment for Database Security

- Start before a database is created
- Separate environments are needed for the development of software, testing, and for production

The Database Administrator Role (DBA)

- An abstract user login should be created as the database administrator.
- The DBA login should be the creator and owner of the production database and tables.
- The DBA account should be the only user with alter privileges on any table in the production database.
- As the owner of the database and all tables, the DBA will control and grant privileges to other users.
- Or – have a separate Database Security Role which manages users and privileges.

Informix Database Users

- Operating System Users - Default and most Common
- Other Options:
  - Mapped Users from:
    - Kerberos single sign-on (SSO)
    - Pluggable Authentication Module (PAM)
  - Internal Users

Steps to Create Internal Users on Unix/Linux

As root or System Administrator:

- Create a default Template OS User for Home Directory and Shell

```
adduser -s /sbin/nologin ifxuser
```

- Identify the Default User in the file: /etc/informix/allowed.surrogates

```
cat /etc/informix/allowed.surrogates
#Surrogate IDs
USERS:ifxuser
```

Steps to Create Internal Users on Unix/Linux

As Informix:

- Update ONCONFIG USERMAPPING

  onmode -wf USERMAPPING=BASIC

- Update the Informix Cache for Surrogates

  onmode -cache surrogates

- In the Sysuser Database create a default user with the Surrogates OS User

  database sysuser;

  create default user with properties user ifxuser;

Steps to Create Internal Users on Unix/Linux

- Add Non OS users to the Sysuser database

```
database sysuser;
create user lester with password "abcabc";
create user tom with password "abcabc";
create user mike with password "abcabc";
```

- Grant the privileges for other databases

```
database benchmark1;
grant connect to lester, tom, mile;
```

Demo - Create Internal Users on Unix/Linux

Questions?

Send follow-up questions to
Lester@advancedatatools.com

Advanced DataTools

International Informix User Group: http://www.iiug.org

IIUG and IBM announce Informix v.14.10.xC4W1 Technical Deep Dive webcast series!

- CSDK and IHQ - July 29, 2020 at 10 am Central (July 29, 2020 15:00 GMT)
- Replication - August 12, 2020 at 10 am Central (August 12, 2020 15:00 GMT)
- Java and System Administration - August 26, 2020 at 10 am Central (August 26, 2020 15:00 GMT)

More Info - https://www.iiug.org/events/

Free Informix Tutorials Webcasts

from the IBM Informix Champions

A step by step guide to using Informix Database Servers

- **Getting Started with Informix by Lester Knutsen – January Replay**
- **Configuring a New Informix Server by Lester Knutsen – February Replay**
- **Managing Informix Disk Space – March Replay**
- **Managing Informix Logs – April Replay**
- **Informix Backup, Recovery, and High Availability – May Replay**
- **Connecting Users to Informix Servers – June Replay**
- **Creating Databases and Tables in Informix – July Replay**
- **Basic Informix Server Monitoring – August 20, 2020 at 2:00 pm EDT**
  - **Onstat, Oncheck, and Informix HQ**

Registration and more information: https://advancedatatools.com/tech-info/next-webcasts/

Advanced DataTools

Informix Training
Updated for Informix 14.10

- Attend classes online on the web.
- All you need is:
  - Web browser to connect to our WebEx training system
  - An SSH client (like Putty) to connect to our training lab for hands-on
- Each student uses an 8-core Linux server with 16GB RAM, SSD drives with Informix 14, and several large databases for benchmark exercises.

➢ ~~**May 18-21, 2020 - Informix for Database Administrators DONE**~~

➢ ~~**July 6-9, 2020 - Advanced Informix Performance Tuning DONE**~~

➢ **October 5-8, 2020 - Informix for Database Administrators**

More information and registration at:
https://advancedatatools.com/training/

Informix 14.X Training

Are you ready to take your DBA skills to the next level?

Each student in class will have a server running Informix 14.10 with:

- 8 CPU Cores
- 16 GB RAM
- 1 SSD Disk
- 1-4 Disks

Class size is limited to 8 students.

Attend online using our remote learning system!

Advanced DataTools

Informix Support and Training from the Informix Champions!

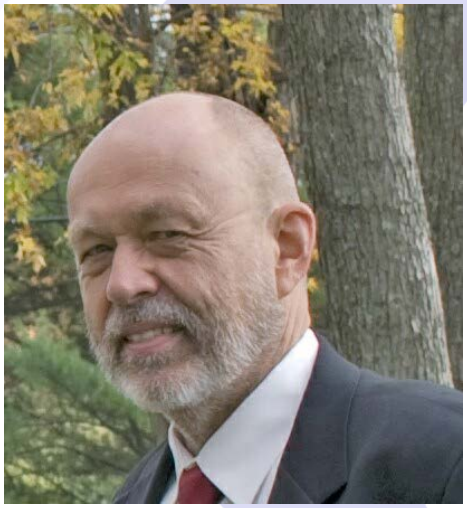
Advanced DataTools is an Advanced Level IBM Informix Data Management Partner, and has been an authorized Informix partner since 1993. We have a long-term relationship with IBM, we have priority access to high-level support staff, technical information, and Beta programs. Our team has been working with Informix since its inception, and includes 8 Senior Informix Database Consultants, 4 IBM Champions, 3 IIUG Director's Award winners, and an IBM Gold Consultant. We have Informix specialists Lester Knutsen and Art Kagel available to support your Informix performance tuning and monitoring requirements!

- ***Informix Remote DBA Support Monitoring***
- ***Informix Performance Tuning***
- ***Informix Training***
- ***Informix Consulting***
- ***Informix Development***

Free Informix Performance Tuning Webcast replays at:
https://advancedatatools.com/tech-info/next-webcasts/
Email: info@advancedatatools.com
Web: https://www.advancedatatools.com

Thank You

Advanced DataTools Corporation



For more information:

Lester@advancedatatools.com

https://www.advancedatatools.com

Advanced DataTools