Hidden Gems of IBM i

Scott Forstie – Business Architect, Db2 for i
forstie@us.ibm.com
@Forstie_IBMi
Finding a “Hidden Gem”

• Every release = 100s of new functions
  • Some big
  • Some small
• How to know them all?
Finding a “Hidden Gem”

- Every release = 100s of new functions
  - Some big
  - Some small
- How to know them all?
- All IBM users have favourites
- These are some of our...
“Hidden Gems”
Database – Create or Replace Tables
Create OR REPLACE Table

Data Definition Language (DDL) SQL statements that support the optional ‘OR REPLACE’ clause:

- CREATE OR REPLACE ALIAS
- CREATE OR REPLACE FUNCTION
- CREATE OR REPLACE MASK
- CREATE OR REPLACE PERMISSION
- CREATE OR REPLACE PROCEDURE
- CREATE OR REPLACE SEQUENCE
- CREATE OR REPLACE TABLE
- CREATE OR REPLACE TRIGGER
- CREATE OR REPLACE VARIABLE
- CREATE OR REPLACE VIEW

Replacing a table:
- Data-Centric
- Dependent Views & MQTs preserved
- Triggers preserved
- RCAC controls preserved
- Auditing preserved
- Authorizations preserved
- Comments and Labels preserved
- Rows optionally deleted

Knowledge Center


Article for previous OR REPLACE statements

http://iprodeveloper.com/database/use-sql-create-or-replace-improve-db2-i-object-management
Create OR REPLACE Table

- CREATE OR REPLACE TABLE allows users to manage the master table source.
- The attributes specified on the CREATE OR REPLACE TABLE will be compared to the existing attributes and the corresponding alters are performed.

You Build it

```
ALTER TABLE corpdata.employee
ALTER COLUMN firstnme
    SET DATA TYPE VARCHAR(20) NOT NULL
ALTER COLUMN lastname
    SET DATA TYPE VARCHAR(30) NOT NULL
ALTER COLUMN phoneno
    SET DATA TYPE VARCHAR(13)
ADD COLUMN level INT BEFORE edlevel;
```

Db2 for i management

```
CREATE OR REPLACE TABLE corpdata.employee(
    empno CHAR(6) NOT NULL,
    firstnme VARCHAR(20) NOT NULL,
    midinit CHAR(1) NOT NULL,
    lastname VARCHAR(30) NOT NULL,
    workdept CHAR(3) DEFAULT NULL,
    phoneno VARCHAR(13) DEFAULT NULL,
    hiredate DATE DEFAULT NULL,
    job CHAR(8) DEFAULT NULL,
    level INT,
    edlevel SMALLINT NOT NULL,
    sex CHAR(1) DEFAULT NULL,
    birthdate DATE DEFAULT NULL,
    salary DECIMAL(9, 2) DEFAULT NULL,
    bonus DECIMAL(9, 2) DEFAULT NULL,
    comm DECIMAL(9, 2) DEFAULT NULL,
    PRIMARY KEY(empno)
) ;
```
Create OR REPLACE Table

Db2 for i implements table replacement using the necessary set of ALTER operations. If alter doesn’t support the action, neither will create or replace table.

Usage Question: **Do you want to preserve the data?**

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DELETE ROWS</strong></td>
<td>All rows are deleted, no delete triggers are fired</td>
</tr>
<tr>
<td><strong>PRESERVE ALL ROWS</strong> <em>(default)</em></td>
<td>Rows are always preserved, columns can be dropped or altered</td>
</tr>
<tr>
<td><strong>PRESERVE ROWS</strong></td>
<td>Rows are preserved, unless a range is eliminated from a partitioned table, if a specified range or partition name matches, the partition is preserved, columns can be dropped or altered</td>
</tr>
</tbody>
</table>
Create OR REPLACE Table

What about CREATE TABLE AS or CREATE TABLE LIKE?

CREATE OR REPLACE TABLE EMPLOYEE AS
(SELECT * FROM MASTER_TABLES.EMPLOYEE)
WITH NO DATA
INCLUDING IDENTITY COLUMN ATTRIBUTES
INCLUDING COLUMN DEFAULTS
INCLUDING IMPLICITLY HIDDEN COLUMN ATTRIBUTES
INCLUDING ROW CHANGE TIMESTAMP COLUMN ATTRIBUTES
ON REPLACE PRESERVE ROWS;

CREATE OR REPLACE TABLE EMPLOYEE LIKE
MASTER_TABLES.EMPLOYEE
INCLUDING IDENTITY COLUMN ATTRIBUTES
INCLUDING COLUMN DEFAULTS
INCLUDING IMPLICITLY HIDDEN COLUMN ATTRIBUTES
INCLUDING ROW CHANGE TIMESTAMP COLUMN ATTRIBUTES
ON REPLACE PRESERVE ROWS;

Using CREATE TABLE AS
- Copy-options can be used to retain columns and attributes
- Constraints are not included
- Must use WITH NO DATA

Using CREATE TABLE LIKE
- Copy-options can be used to retain columns and attributes
- Constraints are not included
Create OR REPLACE Table

How does dependency management work?

CREATE OR REPLACE TABLE DEMO_IT (
    FRST CHAR(6) CCSID 37 NOT NULL,
    SCND INTEGER,
    THRD VARCHAR(10)
)

CREATE OR REPLACE VIEW VIEW_IT AS
    SELECT * FROM DEMO_IT
CREATE INDEX INDEX_IT ON DEMO_IT(THRD);

CREATE OR REPLACE TABLE DEMO_IT (
    FIRST_NAME FOR COLUMN FRST CLOB(1K) NOT NULL,
    SECOND_NAME FOR COLUMN SCND BIGINT DEFAULT -1,
    THIRD_NAME FOR COLUMN THRD VARCHAR(1000)
)

Dependent object management:
- Column names (SQL names), data types and attribute changes are reflected in dependent objects
- System column names (field names) cannot be changed
- If Db2 for i cannot gain exclusive access to all the dependent objects, the operation will fail with SQL0913
- If the change is incompatible, the operation will fail

VIEW_IT field definitions before & after the replacing the table
Create OR REPLACE Table

Generating DDL for existing tables will normally produce separate statements for the table and its constraints. Use the GENERATE_SQL() procedure to produce master table source.

CALL qsys2.generate_sql ( ‘EMPLOYEE’, 'TOYSTORE_MINNESOTA_1', 'TABLE', CREATE_OR_REPLACE_OPTION =>'1', CONSTRAINT_OPTION =>'2');

CREATE OR REPLACE TABLE TOYSTORE_MINNESOTA_1.EMPLOYEE ( EMPNO CHAR(6) CCSID 37 NOT NULL, FIRSTNAME VARCHAR(12) CCSID 37 NOT NULL, MIDINIT CHAR(1) CCSID 37 NOT NULL, LASTNAME VARCHAR(15) CCSID 37 NOT NULL, JRDEPT CHAR(3) CCSID 37 DEFAULT NULL, PHONENO CHAR(4) CCSID 37 DEFAULT NULL, HIREDATE DATE DEFAULT NULL, JOB CHAR(8) CCSID 37 DEFAULT NULL, EDLEVEL SMALLINT NOT NULL, SEX CHAR(1) CCSID 37 DEFAULT NULL, BIRTHDATE DATE DEFAULT NULL, SALARY DECIMAL(9, 2) DEFAULT NULL, BONUS DECIMAL(9, 2) DEFAULT NULL, COMMA DECIMAL(9, 2) DEFAULT NULL, CONSTRAINT TOYSTORE_MINNESOTA_1.Q_TOYST00001_EMPLOYEE_EMPNO_00001 PRIMARY KEY( EMPNO ), CONSTRAINT TOYSTORE_MINNESOTA_1.RED FOREIGN KEY( JRDEPT ) REFERENCES TOYSTORE_MINNESOTA_1.DEPARTMENT( DEPTNO ) ON DELETE SET NULL ON UPDATE NO ACTION CONSTRAINT TOYSTORE_MINNESOTA_1.NUMER CHECK( PHONENO >= '0000' AND PHONENO <= '9999') );
Managing Database Changes in Production
Fair Lock Option

**Challenge:** Frequent DML activity blocks DDL request

**Response:** `PREVENT_ADDITIONAL_CONFLICTING_LOCKS` QAQQINI control

**Benefit:** Improved ability to transform data model in production

**Support:** Applies to ALTER TABLE (Add, Alter or Drop Column), CREATE TRIGGER, LOCK TABLE, & RENAME TABLE
Fair Lock Option

**Challenge:** Seemingly impossible to make DDL changes in production

**Response:** `ALLOW_DDL_CHANGES_WHILE_OPEN` QAQQINI control

**Benefit:** Ability to deploy trigger changes without quiescing user activity

**Support:** Applies to CREATE TRIGGER, ALTER TRIGGER, DROP TRIGGER, COMMENT ON TRIGGER, and LABEL ON TRIGGER, ADDPFTRG, RMVPFTRG, and CHGPFTRG

The *existing* trigger program is being used.
Fair Lock Option

Challenge: Seemingly impossible to make DDL changes in production

Response: ALLOWDDLCHANGESWHILEOPEN QAQQINI control

Benefit: Ability to deploy trigger changes without quiescing user activity

Support: Applies to CREATE TRIGGER, ALTER TRIGGER, DROP TRIGGER, COMMENT ON TRIGGER, and LABEL ON TRIGGER, ADDPFTRG, RMVPFTRG, and CHGPFTRG

The trigger program is being replaced

The existing trigger program is being used
Fair Lock Option

**Challenge**: Seemingly impossible to make DDL changes in production

**Response**: `ALLOW_DDL_CHANGES WHILE OPEN` QAQQINI control

**Benefit**: Ability to deploy trigger changes without quiescing user activity

**Support**: Applies to `CREATE TRIGGER`, `ALTER TRIGGER`, `DROP TRIGGER`, `COMMENT ON TRIGGER`, and `LABEL ON TRIGGER`, `ADDPFTRG`, `RMVPFTRG`, and `CHGPFTRG`
Fair Lock Option

Challenge: Seemingly impossible to make DDL changes in production
Response: `ALLOW_DDL_CHANGES_WHILE_OPEN` QAQQINI control
Benefit: Ability to deploy trigger changes without quiescing user activity
Support: Applies to CREATE TRIGGER, ALTER TRIGGER, DROP TRIGGER, COMMENT ON TRIGGER, and LABEL ON TRIGGER, ADDPFTRG, RMVPFTRG, and CHGPFTRG

The old trigger program is automatically deleted when there are no users.

The new trigger program is used for any new triggering events.
Fair Lock Option

- Using the QAQQINI (Query Options) control
  -- CHGQRYA
  -- OVERRIDE_QAQQINI

```
call qsys2.override_qaqqini(1, '', '');
call qsys2.override_qaqqini(2,
  'ALLOW_DDL_CHANGES_WHILE_OPEN',
  '*YES');
```

```
CREATE OR REPLACE TRIGGER toystore.new_hire
  AFTER INSERT ON toystore.employee
  FOR EACH ROW MODE DB2SQL
  UPDATE toystore.company_stats
  SET Number_of_employees = Number_of_employees + 1;
```

Dawn May – “i Can” Blog – “Managing Trigger Programs in Production”

Implicit Remote Database Access
Implicit Remote Database Access

- A local application can run SQL statements against a local database or a remote database.

- To specify a remote database, you can use a three-part name. A three-part name consists of the RDB name, schema/library name, and object name.
  - SQL naming: <database-name>.<schema-name>.<object-name>
  - System naming: <database-name>/<schema-name>/<object-name>

```plaintext
CL: ADDRDBDIRE RDB(X1423P2) RMTLOCNAME(X1423P2 *IP);

CREATE TABLE X1423P2.TOYSTORE.EMPLOYEE (EMPNO CHAR(6),
                                          FIRSTNME CHAR(10),
                                          LASTNAME CHAR(15));

INSERT INTO X1423P2.TOYSTORE.EMPLOYEE
VALUES ('000002','Michael','Thompson');

SELECT * FROM X1423P2.TOYSTORE.EMPLOYEE;
```
Implicit Remote Database Access

Oracle and SQL Server do not support DRDA as a Application Server

Application Requestor (AR)

- IBM Db2 for i
- IBM Db2 for z/OS
- IBM Db2 for Linux, UNIX and Windows (iLW)
- Other Db2® database products
- IBM Informix
- Other databases (check your database vendor for their DRDA support statement)

Application Server (AS)

- IBM Db2 for i
- IBM Db2 for z/OS
- IBM Db2 for Linux, UNIX and Windows (LUW)
- Other Db2® database products
- Other databases (check your database vendor for their DRDA support statement)

---

Article: Improve Your Data Center with Three-part Name Aliases

Article: Achieve improved database interoperability with SQL and RDB aliases
CREATE TABLE with remote SUBSELECT

- CREATE TABLE AS allows the select to reference a single remote database
- Db2 for i recognizes the remote connection and implicitly manages the connection

```sql
CREATE TABLE DATALIB.MY_TEMP_TABLE (SERVER_NAME, DATA_VALUE)
AS (SELECT CURRENT_SERVER CONCAT ' is the Server Name', IBMREQD
    FROM X1423P2.SYSIBM.SYSDUMMY1) WITH DATA;

SELECT * FROM DATALIB.MY_TEMP_TABLE;
```
CREATE TABLE with remote SUBSELECT

- Use of an ALIAS is the best practice for remote 3-part names because it shields the application. (database transparency)
- Notice how the text of the query does not change

```
CREATE OR REPLACE ALIAS DATALIB.TARGET_TABLE FOR X1423P2.SYSIBM.SYSDUMMY1;

CREATE TABLE DATALIB.MY_TEMP_TABLE(Server_Name) AS
    (SELECT CURRENT_SERVER CONCAT ' is the Server Name'
     FROM DATALIB.TARGET_TABLE)
    WITH DATA;

CREATE OR REPLACE ALIAS DATALIB.TARGET_TABLE FOR LP01UT18.SYSIBM.SYSDUMMY1;

INSERT INTO DATALIB.MY_TEMP_TABLE
    (SELECT CURRENT_SERVER CONCAT ' is the Server Name'
     FROM DATALIB.TARGET_TABLE);

SELECT * FROM DATALIB.MY_TEMP_TABLE;
```
RDB alias support for 3-part SQL statements

- Instead of using CREATE ALIAS (SQL) to deploy database transparency, the Relational Database Directory Entry Alias name can be used.

```
ADDRDBDIRE RDB(X1423P2 MYALIAS) RMTLOCNAME(X1423P2 *IP)
INSERT INTO WORKTABLE SELECT * FROM MYALIAS.TOYSTORE.EMPLOYEE

CHGRDBDIRE RDB(LP13UT26 MYALIAS) RMTLOCNAME(LP13UT26 *IP)
INSERT INTO WORKTABLE SELECT * FROM MYALIAS.TOYSTORE.EMPLOYEE
```

Note:
The SQL statement text does not change
System Limits
System Limits

**Customer Requirements**

- We need to be **proactive** and understand our posture against important system limits
- I want to be able to **recognize** trends and run-away situations
- We need to **understand** how spikes like month-end processing affect our consumption of operating system resources.

**IBM i Innovation**

- **Leverage** the integrated IBM i operating system to instrument the automated recognition of resource consumption
- Accommodate **different types** of consumption (Job, Object, ASP, and System)
- **Db2 for i** is the repository
- Define the **criteria** for which limits are worthy of tracking

*Patent filed March/2013 → “Integrated Limits Tracking, Trending, and Reporting”*
We need to be **proactive** and understand our posture against important system limits.

I want to be able to **recognize** trends and run-away situations.

We need to **understand** how spikes like month-end processing affect our consumption of operating system resources.

**Leverage** the integrated IBM i operating system to instrument the automated recognition of resource consumption.

Accommodate **different types** of consumption (Job, Object, ASP, and System).

**Db2 for i** is the repository.

Define the **criteria** for which limits are worthy of tracking.

---

*Patent filed March/2013 → "Integrated Limits Tracking, Trending, and Reporting"*
System Limits Architecture

User Job – Long running data purge running with commitment control

10,000th row deleted

110,000th row deleted

210,000th row deleted

Low priority QDBSRVnn jobs

System event notification

INSERT into…
# System Limits – Where does the data reside

<table>
<thead>
<tr>
<th>Object</th>
<th>Type</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>QSYS2/SYSLIMTBL</td>
<td>*FILE SQL Table</td>
<td>System wide (including iASP) physical file repository for tracked System Limits. Designed to have the smallest storage footprint.</td>
</tr>
<tr>
<td>QSYS2/GET_JOB_INFO</td>
<td>User Defined Table Function</td>
<td>Accepts a job name as input and returns a single row of information about an active job.</td>
</tr>
<tr>
<td>QSYS2/SQL_SIZING</td>
<td>*FILE SQL Table</td>
<td>Table where architected limits are defined, including translated descriptions.</td>
</tr>
<tr>
<td>QSYS2/SYSLIMITS</td>
<td>*FILE SQL View</td>
<td>The external interface which joins detail from the preceding three resources.</td>
</tr>
</tbody>
</table>
# System Limits – Documentation

www.ibm.com/support/knowledgecenter/ssw_ibm_i_73/rzajq/rzajqserviceshealth.htm

## Table 4. Work management limits

<table>
<thead>
<tr>
<th>Limit description</th>
<th>Limit ID</th>
<th>Maximum</th>
<th>Floor</th>
<th>Increment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum number of jobs</td>
<td>19000</td>
<td>970,000</td>
<td>1,000</td>
<td>400</td>
</tr>
<tr>
<td>Maximum number of spool files</td>
<td>19002</td>
<td>2,610,000</td>
<td>10,000</td>
<td>5,000</td>
</tr>
<tr>
<td>Maximum number of spooled files in each independent ASP</td>
<td>19003</td>
<td>10,000,000</td>
<td>10,000</td>
<td>5,000</td>
</tr>
</tbody>
</table>
WITH TT(JOB_MAXIMUM) 
    AS (SELECT CURRENT_NUMERIC_VALUE 
        FROM QSYS2.SYSTEM_VALUE_INFO 
        WHERE SYSTEM_VALUE_NAME = 'QMAXJOB') 
SELECT LAST_CHANGE_TIMESTAMP AS INCREMENT_TIME, CURRENT_VALUE AS JOB_COUNT, 
    TT.JOB_MAXIMUM, DEC(DEC(CURRENT_VALUE,19,2) / DEC(TT.JOB_MAXIMUM,19,2) * 
    100,19,2) AS PERCENT_CONSUMED 
FROM QSYS2.SYSLIMITS, TT 
WHERE LIMIT_ID = 19000 ORDER BY CURRENT_VALUE DESC
Deleting data while under Commitment Control

<table>
<thead>
<tr>
<th>SIZING_NAME</th>
<th>CURRENT_VALUE</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>10000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>10000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>11000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>21000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>21000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>31000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>31000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>41000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>41000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>51000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>51000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>61000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>61000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>71000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>71000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>81000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>81000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>91000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>91000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>10000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROWS LOCKED IN A UNIT OF WORK</td>
<td>10000</td>
</tr>
<tr>
<td>MAXIMUM NUMBER OF ROW CHANGE OPERATIONS IN A UNIT OF WORK</td>
<td>10000</td>
</tr>
</tbody>
</table>

Floor

Increments reflect increasing number of deleted rows

Commit or Rollback releasing the locks

Note... we deleted 1 million rows, the high point is not recorded
Protection automated with a Trigger

- Built into ACS
- Insert from Examples...
## Integrated File System

### Added in IBM i 7.2

<table>
<thead>
<tr>
<th>Limit description</th>
<th>Limit ID</th>
<th>Maximum</th>
<th>Floor</th>
<th>Increment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of objects linked in a directory</td>
<td>18402</td>
<td>0</td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td>Maximum number of directories linked in a directory</td>
<td>18403</td>
<td>1,000,000</td>
<td>1,000</td>
<td>1,000</td>
</tr>
<tr>
<td>Maximum number of file system objects in *SYSBAS ASPs</td>
<td>18404</td>
<td>2,147,483,647</td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td>Maximum number of file system objects in an independent ASP</td>
<td>18405</td>
<td>2,147,483,647</td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td>Maximum number of document library objects in a folder</td>
<td>18406</td>
<td>65510</td>
<td>1,000</td>
<td>500</td>
</tr>
<tr>
<td>Number of document library objects in the system ASP</td>
<td>18407</td>
<td>0</td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td>Maximum number of document library objects in a user ASP</td>
<td>18408</td>
<td>1,000,000</td>
<td>100,000</td>
<td>10,000</td>
</tr>
<tr>
<td>Maximum number of bytes in a stream file</td>
<td>18409</td>
<td>1,099,511,627,776</td>
<td>16,777,216</td>
<td>1,048,576</td>
</tr>
<tr>
<td>Maximum number of bytes in a document</td>
<td>18410</td>
<td>2,147,483,647</td>
<td>16,777,216</td>
<td>1,048,576</td>
</tr>
</tbody>
</table>
Find the largest IFS files

```
SELECT LASTCHG, JOB_NAME, ASP_NUMBER, IFS_PATH_NAME, USER_NAME, CURRENT_VALUE FROM QSYS2.SYSLIMITS WHERE LIMIT_ID = 18409 ORDER BY CURRENT_VALUE DESC;
```
Set Server Subsystem Routing
This procedure can be used to configure alternate subsystems by user and IBM i server name. This allows an IBM i administrator to relocate users into subsystems that are configured to meet user expectations or to protect overall system resources.

Procedure QSYS2.SET_SERVER_SBS_ROUTING()

Procedure Parameters:

1. Authorization Name
   The user profile name

2. Server Name
   QZDASOINIT, QRWTSRVR, and many others or *ALL

3. Alternate Subsystem Name
   The name of the subsystem to use

4. Allow Rollover (YES or NO)
   If the alternate subsystem cannot be used, should the default subsystem be used or should the connect fail?

Authorization name can be:
- User name
- Group name
- Supplemental Group name
Example…

- Construct a subsystem that will constrain the amount of system resources available to users who are known to execute ad hoc queries.

```cl
CL: CRTSBSD SBSD(QGPL/ADHOCSBS) POOLS((1 *BASE))
   TEXT('Ad hoc users SBS');
CL: CRTJOBQ QGPL/ADHOCJOBQ TEXT('Ad hoc users job queue');
CL: ADDJOBQE SBSD(QGPL/ADHOCSBS) JOBQ(QGPL/ADHOCJOBQ)
   MAXACT(100) SEQNBR(40);
CL: CRTCLS CLS(QGPL/ADHOCCLS) RUNPTY(55) TIMESLICE(100)
   TEXT('Ad hoc class');
-- Repeat the ADDPJE for each server name
CL: ADDPJE SBSD(QGPL/ADHOCSBS) PGM(QSYS/QZDASOINIT)
   JOBD(QGPL/QDFTSVR) CLS(QGPL/ADHOCCLS);
CL: STRSBS SBSD(QGPL/ADHOCSBS);
CL: CALL QSYS2.SET_SERVER_SBS_ROUTING('JOEUSER', '*ALL', 'ADHOCSBS', 'NO');
```
QSYS2.SERVER_SBS_ROUTING – View

- QSYS2.SERVER_SBS_ROUTING is used to access the alternative subsystem user configuration

- The configuration detail is stored within the *USRPRF objects

- **Authorization requirements** to change the configuration:
  *SECADM user special authority
  *OBJMGT and *USE to the target *USRPRF

```sql
SELECT * FROM QSYS2.SERVER_SBS_ROUTING;
```
# Configurable Servers


## Table 1. Servers and default subsystems

<table>
<thead>
<tr>
<th>Server Description</th>
<th>Server Name</th>
<th>Default subsystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Central server</td>
<td>QZSCSRVS</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>Database server</td>
<td>QZDASOINIT</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>Data queue server</td>
<td>QZHQSSRV</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>DDM</td>
<td>QRWTSRVR</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>DRDA</td>
<td>QRWTSRVR</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>File server</td>
<td>QPWFSERVSO</td>
<td>QSERVER</td>
</tr>
<tr>
<td>Network print server</td>
<td>QNPSERVS</td>
<td>QUSRWRK</td>
</tr>
<tr>
<td>Remote command server</td>
<td>QZRCSRVS</td>
<td>QUSRWRK</td>
</tr>
</tbody>
</table>
Grouping similar Navigator users

Manage ACS users
- Avoid having all users run in QUSRWRK, with the same priority
- Setup once and manage the Group Profile

-- Description: Reposition all Navigator users into a controlled subsystem and do not allow connections to fall-over into the default subsystem (QUSRWRK or QSERVER) if the INAVGRP subsystem cannot be used

CALL QSYS2.SET_SERVER_SBS_ROUTING(
  AUTHORIZATION_NAME => 'INAVGRP',
  SERVER_NAME        => '*ALL',
  SUBSYSTEM_NAME     => 'INAVSBS',
  ALLOW_ROLLOVER     => 'NO');
Database Maintenance in System i Navigator
Database Maintenance in System i Navigator

Two primary use cases:
1. Examine history of long running database maintenance operations
2. Monitor active database maintenance operations
Database Maintenance in System i Navigator

- **Index builds**
  - Alters with unique index(es)
  - Reorganize
  - New index create

- **Index rebuilds**
  - Restored base table without restoring the index
  - Alters with non-unique index(es)

See the progress...ETA is a guess

If an anomaly occurred...
check the status file before repeating
Access Client Solutions – Integrated File System
IBM i Access Client Solutions – Integrated File System

Integrated File System provides an interface for browsing the integrated file system of your IBM i and working with objects in the integrated file system in the following ways:

- Create new folders (directories, libraries, and folders)
- Delete objects
- Rename objects
- Copy objects between your client system and your IBM i
- Copy objects within the integrated file system on an IBM i or to the integrated file system on another IBM i
- Send objects to another IBM i (or to several)
- View properties of objects

Note: Copying or sending objects from one IBM i to another uses the QFileSvr.400 file system.

This task requires a system configuration. To add or change a system configuration, select System Configurations from the Management tasks.
IBM i Access Client Solutions – Integrated File System
Thank You
Reorganize Physical File Member

Tips and tricks for expediting reorganizations
• DB2 Symmetric Multiprocessing (SMP) – Parallel reorganize and index builds
• Use database catalogs to assess the need for reorganizes and the best strategy

Relevant enhancements in IBM i 7.2, 7.3 and future
• Database Reorganization – User specified starting point
• Honor priority change for parallel index build
• Enhanced index build logic for highly concurrent environments
• QSYS2.SYSLIMITS
• System i Navigator’s Database Maintenance support
Reorganize Physical File Member
Reorganize Physical File Member

```sql
--
-- Identify candidates for physical file reorganization
-- Examine files with more than a million rows deleted
--
SELECT table_schema,
       TABLE_NAME,
       number_rows AS valid_rows,
       number_deleted_rows AS deleted_rows,
       data_size AS data_space_size_in_bytes,
       DEC(DEC(number_deleted_rows, 19, 2) /
       DEC(number_rows + number_deleted_rows, 19, 2) * 100, 19, 2) AS deleted_row_percentage
FROM dbestudy.toystore_tables_runtime_details a
WHERE number_deleted_rows > 1000000
ORDER BY deleted_row_percentage DESC;
```
Reorganize Physical File Member

--
-- Review the distribution of deleted records
--

```
SELECT 1000000 - COUNT(*) AS DELETEDCNT 
FROM item_fact A 
GROUP BY BIGINT(RRN(A) / 1000000) 
ORDER BY BIGINT(RRN(A) / 1000000)
```
<table>
<thead>
<tr>
<th></th>
<th>ALWCANCEL(*NO)</th>
<th>ALWCANCEL(*YES)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>KEYFILE (&quot;NONE&quot;)</td>
<td>KEYFILE &quot;FILE or keyfile&quot;</td>
</tr>
<tr>
<td>Cancel and restart</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Concurrent Access</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Parallel processing</td>
<td>Only index rebuilds</td>
<td>Only index rebuilds</td>
</tr>
<tr>
<td>Non-parallel performance</td>
<td>Very fast</td>
<td>Fast</td>
</tr>
<tr>
<td>Temporary storage</td>
<td>Double data storage</td>
<td>Double data storage</td>
</tr>
<tr>
<td>LIFO KEYFILE index processing</td>
<td>N/A</td>
<td>Duplicates reversed</td>
</tr>
<tr>
<td>Index processing (non-KEYFILE)</td>
<td>Synchronous or asynchronous rebuilds</td>
<td>Synchronous or asynchronous rebuilds</td>
</tr>
<tr>
<td>Final row position exact</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Amount of CPU and I/O used</td>
<td>Smallest</td>
<td>Next smallest</td>
</tr>
<tr>
<td>Variable length segment reorganize</td>
<td>Good</td>
<td>Good</td>
</tr>
<tr>
<td>Allows referential integrity parents and FILE LINK CONTROL DataLinks</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Allows QTEMP and Database Cross Reference Files</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>HABP replication cost</td>
<td>Minimal - one journal entry</td>
<td>Minimal - one journal entry</td>
</tr>
</tbody>
</table>
Database – Constraints
Gems you’ve owned for decades…

- Data-Centric technologies save you time and money

**Long Time Gems**

- SQL Views ➔ Dawn of time
- Primary Keys ➔ V3R1M0
- Foreign Keys ➔ V3R1M0
- Native Triggers ➔ V3R1M0
- Check Constraints ➔ V3R6M0
- SQL Triggers ➔ V5R1M0
More Recently Added...

- Data-Centric technologies save you time and money

- Field Procedures
- Implicitly hidden columns
- Range and Hash Partitioning
- Row Change Timestamp
- And more...

- Row Permissions
- Column Masks
- Media Preference
- Memory Preference
- EVI Only Access

- Temporal Tables
- Generated Columns for Auditing
- New OLAP Specifications

Constraints

- **Constraints** enforce the business rules defined by the data model

There are three types of constraints:

1. A **unique constraint** is a rule that forbids duplicate values in one or more columns within a table.
   
   Two forms:
   
   a) **Unique Key(s)** – a unique index is used
   
   b) **Primary Key** – a single column with a unique, non-NULL value
   
   (sometimes an Identity value is used)

2. A **referential constraint** is a logical rule about values in one or more columns in one or more tables

3. A **check constraint** sets restrictions on data added to a specific table
Constraints Enforce the Rules

Unique Keys
Provide Single Row Retrieval
Constraints Enforce the Rules

- Unique Keys
  - Provide Single Row Retrieval

- Primary Keys
  - Used to Establish Relationships
Constraints Enforce the Rules

Unique Keys

Provide Single Row Retrieval

Primary Keys

Used to Establish Relationships

Foreign Keys

Connect the Dots
Constraints Enforce the Rules

Primary Keys
Used to Establish Relationships

Foreign Keys
Connect the Dots

Check Constraint
Must be 'M' or 'F'

Unique Keys
Provide Single Row Retrieval

Primary Keys
Used to Establish Relationships

Constraints Enforce the Rules

Check Constraint
Must be 'M' or 'F'

Unique Keys
Provide Single Row Retrieval

Primary Keys
Used to Establish Relationships

Foreign Keys
Connect the Dots

Constraints Enforce the Rules

Check Constraint
Must be 'M' or 'F'

Unique Keys
Provide Single Row Retrieval

Primary Keys
Used to Establish Relationships

Foreign Keys
Connect the Dots