What’s new in Db2 for i

Sarah Mackenzie
Db2 for i, Senior Software Engineer
samacken@us.ibm.com

Database enhancements with:
IBM i 7.5 TR1
IBM i 7.4 TR7

Announced → October 11, 2022
General Availability → December 2, 2022

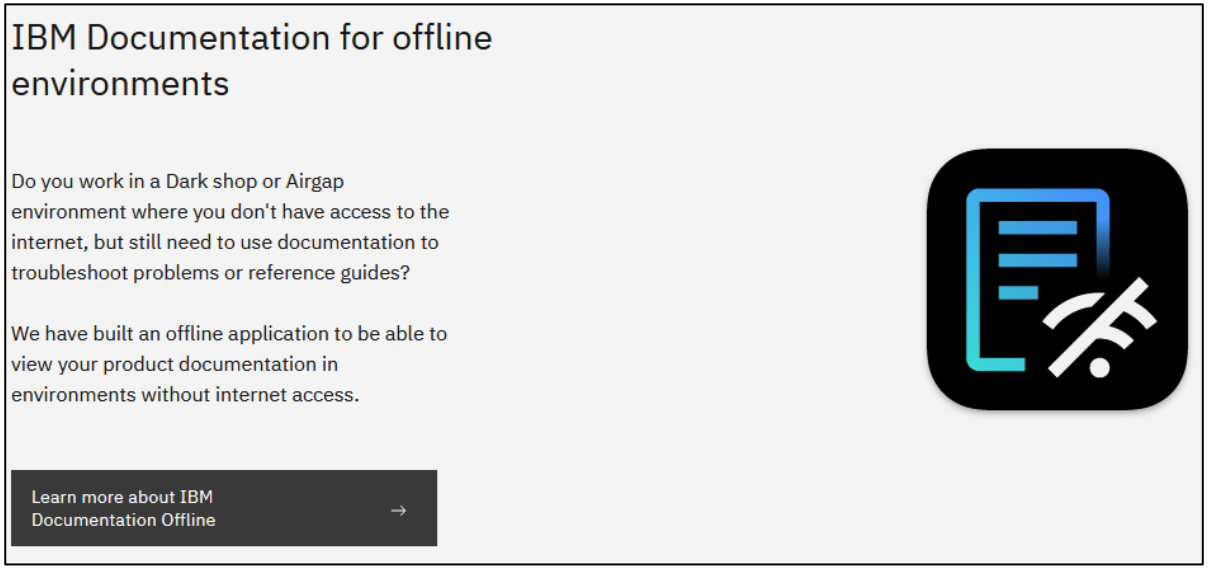
	Enhancement Landing Pages
IBM i 7.5	TR1 - Base Enhancements
IBM i 7.4	TR7 - TR6 - TR5 - TR4 - TR3 - TR2 - TR1 - Base Enhancements
IBM i 7.3	TR11 - TR10 - TR9 - TR8 - TR7 - TR6 - TR5 - TR4 -TR3 - TR2 -TR1 - Base Enhancements
IBM i 7.2	TR9 - TR8 - TR7 - TR6 - TR5 - TR4 - TR3 - TR2 - TR1 - Base Enhancements

www.ibm.com/ibmi/techupdates

IBM Documentation

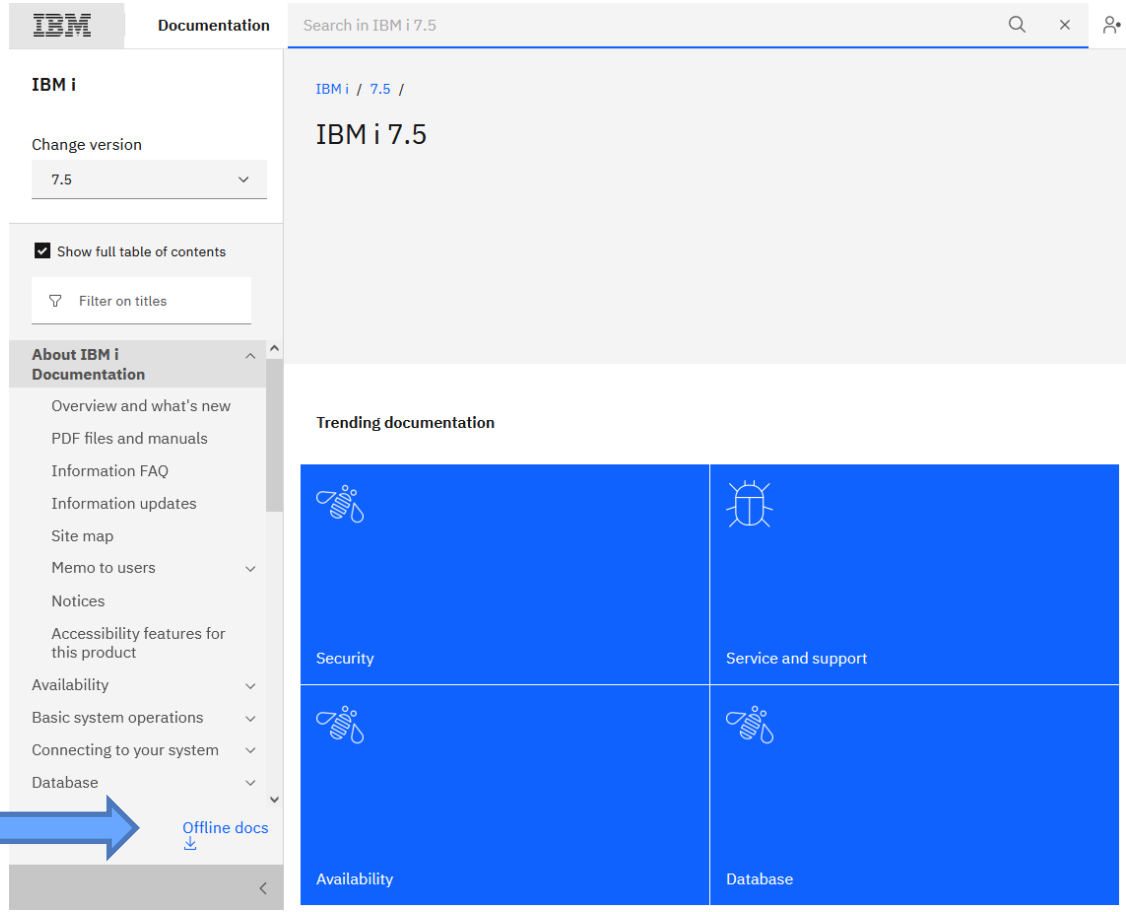
IBM Documentation Offline

- Use IBM Documentation Offline
  - No connection needed
  - Performance advantaged
  - https://www.ibm.com/docs/en/offline



IBM Documentation Offline

- Access Documentation Offline
- Available NOW to Windows users
- IBM i 7.5 offline ~1.8GB
- To see the ‘Offline docs’ download option, you need to be signed in



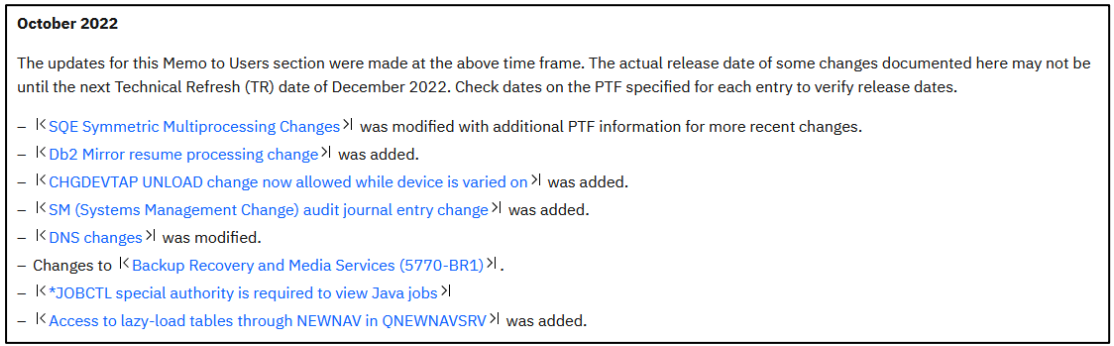
IBM Documentation Offline

- Download the IBM Documentation Offline App
  - https://www.ibm.com/docs/en/offline
- Download IBM Documentation
  - https://www.ibm.com/docs/en/i/7.5
- Add the downloaded documentation to the app

Memorandum To Users

- An important book to read before upgrading the IBM i operating system
- An important book to read before installing PTF Groups

- Direct links to the MTU:
  - IBM i 7.3: https://www.ibm.com/docs/en/i/7.3?topic=users-whats-new
  - IBM i 7.4: https://www.ibm.com/docs/en/i/7.4?topic=users-whats-new
  - IBM i 7.5: https://www.ibm.com/docs/en/i/7.5?topic=users-whats-new



October 2022

The updates for this Memo to Users section were made at the above time frame. The actual release date of some changes documented here may not be until the next Technical Refresh (TR) date of December 2022. Check dates on the PTF specified for each entry to verify release dates.

- SQE Symmetric Multiprocessing Changes was modified with additional PTF information for more recent changes.
- Db2 Mirror resume processing change was added.
- CHGDEVTAP UNLOAD change now allowed while device is varied on was added.
- SM (Systems Management Change) audit journal entry change was added.
- DNS changes was modified.
- Changes to Backup Recovery and Media Services (5770-BR1).
- *JOBCTL special authority is required to view Java jobs
- Access to lazy-load tables through NEWNAV in QNEWNAVSRV was added.

SQL enhancements

What's new for User-defined Tables

- What is a User Defined Table Function (UDTF)?
  - A user-defined function
    - Optional input parameters
    - Has an SQL routine body that has simple to complex logic
    - Returns a set of rows as a table
  - Functions can be referenced in SQL statements
    - UDTFs are only used in the FROM clause of a SELECT statement
  - Many IBM i services are implemented as UDTFs

```
SELECT * FROM TABLE (QSYS2.OBJECT_STATISTICS('MYLIB','JRN'));
```

OBJNAME	OBJTYPE	OBJOWNER	OBJDEFINER	OBJCREATED	OBJSIZE	OBJTEXT	OBJLONGNAME
JRN1	*JRN	SAMACKEN	SAMACKEN	2022-10-17 11:00:41.000000	12288	<NULL>	JRN1
JRN2	*JRN	SAMACKEN	SAMACKEN	2022-10-17 11:00:42.000000	12288	<NULL>	JRN2
JRN3	*JRN	SAMACKEN	SAMACKEN	2022-10-17 11:00:43.000000	12288	<NULL>	JRN3

REMOTE TABLE

- The UDTF can target a different Db2 for i database
  - Using the REMOTE keyword
  - Specifying a three-part name
    - Relational database (RDB) name
    - `SELECT * FROM REMOTESYS.MYLIB.T1;`

```
-- Run locally
SELECT * FROM
  TABLE(QSYS2.OBJECT_PRIVILEGES('MYLIB', 'T1', '*FILE'));
```

```
-- Run remote
SELECT * FROM
  REMOTE TABLE(REMOTESYS.QSYS2.OBJECT_PRIVILEGES('MYLIB', 'T1', '*FILE'));
```

https://www.ibm.com/docs/en/i/7.5?topic=statement-selecting-data-from-remote-database

COMMENT ON COLUMN

- Add a comment on your UDTF return columns

```
-- Create you own UDTFs
CREATE FUNCTION MYLIB.GET_SYSTEM_NAME_FROM_LONG_NAME (tbl_schema VARCHAR(128),
                                                      tbl_name VARCHAR(128))
  RETURNS TABLE(SYSTEM_TABLE_NAME CHAR(10), SYSTEM_TABLE_SCHEMA CHAR(10))
  BEGIN
    RETURN SELECT SYS_TNAME, SYS_DNAME FROM QSYS2.SYSTABLES
            WHERE DBNAME =  tbl_schema AND NAME = tbl_name;
  END;

SELECT * FROM TABLE(MYLIB.GET_SYSTEM_NAME_FROM_LONG_NAME('MYLIB','LONG_LONG_LONG_T1'));
```

SYSTEM_TABLE_NAME	SYSTEM_TABLE_SCHEMA
LONG_00001	MYLIB

```
-- Add a COMMENT to your UDTF return column
COMMENT ON TABLE FUNCTION RETURN COLUMN
  MYLIB.GET_SYSTEM_NAME_FROM_LONG_NAME (SYSTEM_TABLE_NAME IS 'Table Schema',
                                        SYSTEM_TABLE_SCHEMA IS 'Table Name');
```

https://www.ibm.com/docs/en/i/7.5?topic=statements-comment

New built-in scalar functions

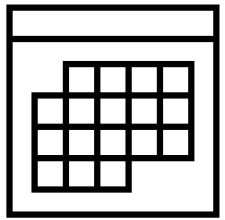
- What is a scalar functions
  - A *scalar function* takes input argument(s) and returns a single value result.
  - A *scalar function* can be used wherever an expression can be used.
- New built-in scalar functions
  - FIRST_DAY
    - Returns a date or timestamp that represents the first day of the month
  - TIMESTAMPDIFF_BIG
    - Returns the difference between two timestamps with added granularity
  - JSON_UPDATE
    - Update or remove a specific value

FIRST_DAY

- Returns a date or timestamp that represents the first day of the month
  - Can be a date, timestamp, character string, or graphic string
  - Similar to LAST_DAY

```
-- Find the first day and last day of the month
SELECT FIRST_DAY(CURRENT_DATE) AS FIRST_DAY,
       LAST_DAY(CURRENT_DATE) AS LAST_DAY
  FROM SYSIBM.SYSDUMMY1;
```

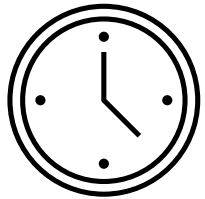
FIRST_DAY	LAST_DAY
2022-10-01	2022-10-31



https://www.ibm.com/docs/en/i/7.5?topic=functions-first-day

TIMESTAMPDIFF_BIG

- Returns an estimated number of intervals between two timestamps
  - You define the intervals
    - Microseconds, second, minutes, hours, days, weeks, months, quarters, years
  - Returns a BIGINT
    - Alternate to TIMESTAMPDIFF
    - Can handle any granularity and scale of timestamp comparison



```
-- Find the elapsed number of microseconds for a job
SELECT JOB_NAME,
       JOB_ACTIVE_TIME,
       TIMESTAMPDIFF_BIG(1, CHAR(JOB_END_TIME-JOB_ACTIVE_TIME)) AS ELAPSED_MICROSECONDS,
FROM TABLE (QSYS2.JOB_INFO(JOB_USER_FILTER=>'SAMACKEN'))
ORDER BY ELAPSED_MICROSECONDS DESC;
```

JOB_NAME	JOB_ACTIVE_TIME	ELAPSED_MICROSECONDS
580023/SAMACKEN/QPADEV0007	2022-08-23 12:58:46	344041000000
752674/SAMACKEN/QPADEV0007	2022-07-29 13:54:09	191436000000
580676/SAMACKEN/QPADEV0008	2022-08-23 15:28:03	184030000000

https://www.ibm.com/docs/en/i/7.5?topic=functions-timestampdiff-big

JSON

- What is JSON?
  - JavaScript Object Notation
  - A format for storing and transporting data
  - Commonly used by web services
- JSON document consist of an object
  - key:value pairs separated by a comma
  - Easy to read and understand
  - Language independent and portable
- JSON support goes back to IBM i 7.2
- **JSON is everywhere**

```
{“FirstName”:”Jane”,
 ”LastName”:”Smith”,
 “Age”:42,
 “Phone”:[{“work”:”555-1234”},
          {“cell”:”555-9887”}]}
```

JSON

- You can build and store JSON objects (existing support)

```
-- Insert JSON data into a table
CREATE TABLE CUSTOMER (CUSTOMER_ID CHAR(5), CUSTOMER_JSON CLOB(2G) CCSID 1208);

INSERT INTO CUSTOMER VALUES
('1S32A', JSON_OBJECT(KEY 'Name' VALUE 'Sarah Mackenzie',
                      KEY 'Address' VALUE '11 Main Street',
                      KEY 'City' VALUE 'Rochester')),
('42K1A', JSON_OBJECT(KEY 'Name' VALUE 'Jane Miller',
                      KEY 'Address' VALUE '1521 Broadway',
                      KEY 'City' VALUE 'Rochester'));

SELECT * FROM CUSTOMER;
```

CUSTOMER_ID	CUSTOMER_JSON
1S32A	{"Name":"Sarah Mackenzie","Address":"11 Main Street","City":"Rochester"}
42K1A	{"Name":"Jane Miller","Address":"1521 Broadway","City":"Rochester"}

JSON_UPDATE

- What is new?
  - JSON_UPDATE
  - Update or remove a value, given a key
  - Options – SET and REMOVE
  - Update address for {"Name":"Sarah Mackenzie', "Address":"11 Main Street','City":"Rochester"}

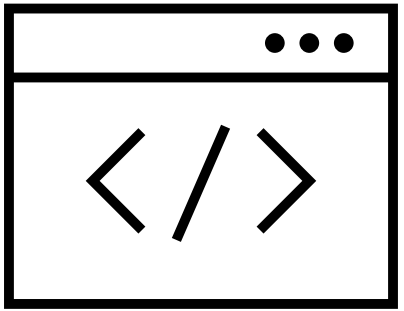
```
-- Update a customer address in a JSON object
UPDATE CUSTOMER
  SET CUSTOMER_JSON = JSON_UPDATE(CUSTOMER_JSON, 'SET', '$.Address','831 Center St')
  WHERE CUSTOMER_ID = '1S32A';

SELECT * FROM CUSTOMER;
```

CUSTOMER_ID	CUSTOMER_JSON
1S32A	{"Name":"Sarah Mackenzie","Address":"831 Center St","City":"Rochester"}
42K1A	{"Name":"Jane Miller","Address":"1521 Broadway","City":"Rochester"}

HTTP functions

- HTTP is the preferred way for communicating on the Web
- RESTful services provide access to information
  - accessed using HTTP
    - addressable via a URL
    - XML and JSON



HTTP functions

- New and improved HTTP functions
  - New functions located in QSYS2 schema
    - Heritage functions in SYSTOOLS
  - Uses the HTTP transport APIs
    - Integrated Web Services (IWS) client for ILE
  - Improved functionality
    - HTTP authentication and proxy support
    - Configurable redirection attempts and SSL options
    - Improved performance
    - Reduced footprint
    - Does not use Java

IBM i 7.5 PTF Group SF99950 Level 1
IBM i 7.4 PTF Group SF99704 Level 15
IBM i 7.3 PTF Group SF99703 Level 26

Connecting to a web service

HTTP functions

- APIs -- HTTP request methods
  - GET
    - Retrieve data from a resource
    - Most common
  - POST
    - Send data to the URL to create or update a resource
    - Data is sent in the request body
    - Another common method
  - PUT
    - Send data to the URL to create or update a resource
    - The same PUT request called multiple times produces the same results
  - DELETE
    - Delete a resource at the URL

HTTP functions

- Verbose or not Verbose
  - Not Verbose
    - Scalar function
    - Return a single response CLOB
  - Verbose
    - Table function
    - Returns a row
      - CLOB response data
      - Header information from the HTTP server
      - JSON
    - Useful for debugging errors

HTTP functions

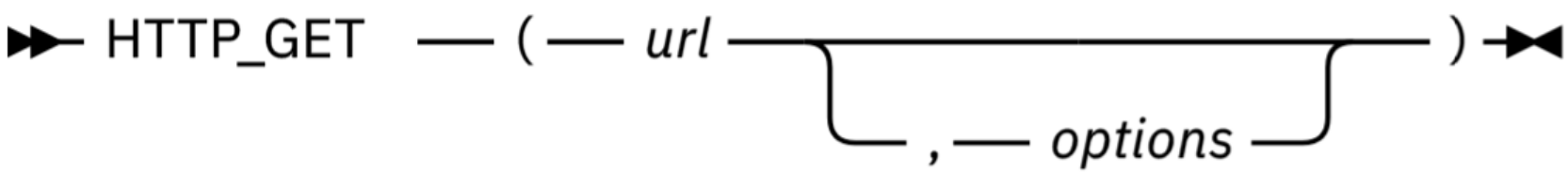
Function	Description
HTTP_DELETE	Make an HTTP DELETE request. Returns CLOB data.
HTTP_DELETE_VERBOSE	Make an HTTP DELETE request. Returns CLOB data and response header.
HTTP_GET	Make an HTTP GET request. Returns CLOB data.
HTTP_GET_VERBOSE	Make an HTTP GET request. Returns CLOB data and response header.
HTTP_POST	Make an HTTP POST request. Return CLOB data.
HTTP_POST_VERBOSE	Make an HTTP POST request. Return CLOB data and response header.
HTTP_PUT	Make an HTTP PUT request. Returns CLOB data.
HTTP_PUT_VERBOSE	Make an HTTP PUT request Returns CLOB data and response header.

HTTP functions

- Enhanced with IBM i 7.5 SF99950 Level 1, IBM i 7.4 SF99704 Level 20, and IBM i 7.3 SF99703 Level 28:

Function	Description
HTTP_PATCH	Make an HTTP PATCH request. Returns CLOB data.
HTTP_PATCH_VERBOSE	Make an HTTP PATCH request. Returns CLOB data and response header.

HTTP_GET



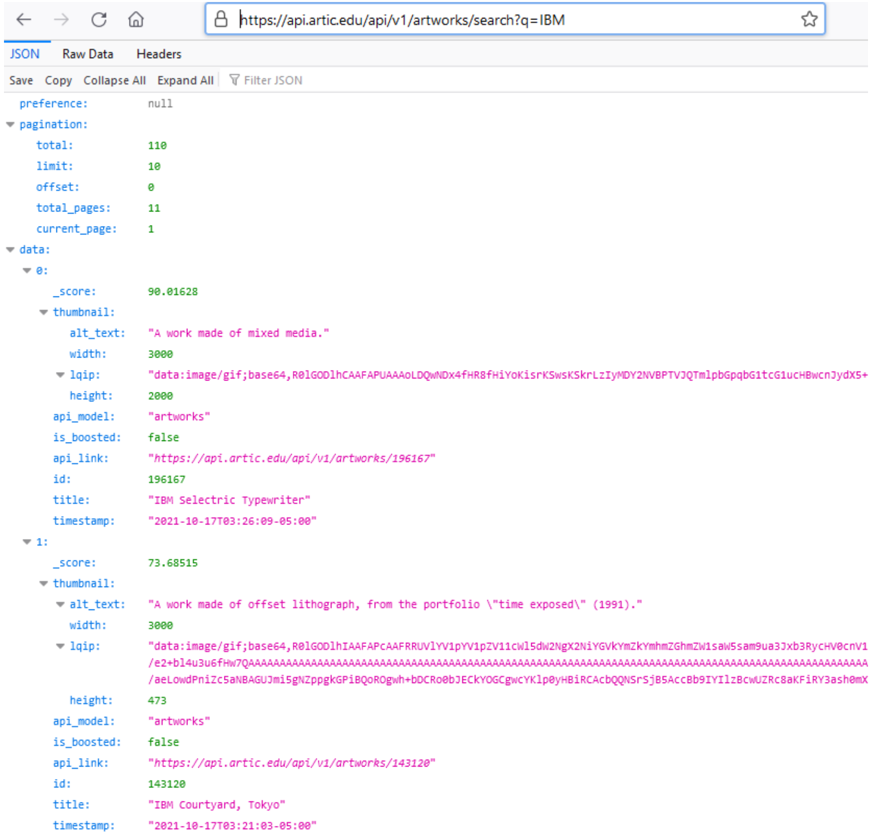
- URL
  - The URL of the resource
- Options
  - JSON
  - Contains the options
    - Headers
    - Basic authentication
    - And many others

HTTP_GET

Art Institute of Chicago API –
Search artwork at the Art Institute

- URL
  - https://api.artic.edu/api/v1/
- Method
  - GET /artworks/search
- Parameter
  - q – Search query

`https://api.artic.edu/api/v1/artworks/search?q=IBM`



HTTP_GET

```
VALUES QSYS2.HTTP_GET(
  'https://api.artic.edu/api/v1/artworks/search?q=IBM',
  '{"sslCertificateStoreFile":"/home/javaTrustStore/fromJava.KDB"}');
```

- Returns CLOB data
  - JSON

HTTP_GET

```
SELECT * FROM
  JSON_TABLE(
    QSYS2.HTTP_GET('https://api.artic.edu/api/v1/artworks/search?q=IBM',
                   '{"sslCertificateStoreFile":"/home/javaTrustStore/fromJava.KDB"}'),
    'lax $.data[*]'
    COLUMNS(alt_text  VARCHAR(100) PATH 'lax $.thumbnail.alt_text',
            title     VARCHAR(100) PATH 'lax $.title'));
```

ALT_TEXT	TITLE
A work made of mixed media.	IBM Selectric Typewriter
A work made of offset lithograph, from the portfolio "time exposed" (1991).	IBM Courtyard, Tokyo
A work made of variable.	IBM PC Convertible
A work made of gelatin silver print.	IBM Building Roof, View East
A work made of chromogenic print.	A. D.'s Place, Glendora, Mississippi
A work made of gelatin silver print.	IBM Building, View South
A work made of chromogenic print.	Rooster at Whispering Pines, Lowndes County, Mississippi
A work made of gelatin silver print.	IBM Building Roof, View Southwest
A work made of oil on canvas.	Correct Me If I'm Wrong
A work made of gelatin silver print.	Big Joe's Funeral, Oktibbeha County, Mississippi

HTTP_GET_VERBOSE

```
SELECT * FROM TABLE(
  QSYS2.HTTP_GET_VERBOSE('https://api.artic.edu/api/v1/artworks/search?q=IBM',
                         '{"sslCertificateStoreFile":"/home/javaTrustStore/fromJava.KDB"}'));
```

RESPONSE_MESSAGE	RESPONSE_HTTP_HEADER
{"preference":null,"pagination":{"total":110,"l...	{"HTTP STATUS CODE":200,"Content-Type":"application/json","Transfer-Encoding":"chunked","Connection":"k...

- Returns table
  - CLOB response data
  - Header information from the HTTP server

```
{"HTTP_STATUS_CODE":200,
 "Content-Type":"application/json",
 "Transfer-Encoding":"chunked",
 "Connection":"keep-alive",
 "Date":"Mon, 17 Oct 2022 20:35:10 GMT",
 "Server":"Apache/2.4.54 () OpenSSL/1.0.2k-fips",
 "Cache-Control":"no-cache, private",
 "Access-Control-Allow-Origin":"*",
 "ETag":"7a599d9004bae1578863587d335d9e7b",
 "Vary":"Accept-Encoding",
 "X-Cache":"Miss from cloudfront",
 "Via":"1.1 4bcf2cf54fd5ae8b72c2c156e5462e44.cloudfront.net (CloudFront)",
 "X-Amz-Cf-Pop":"PHX50-C1",
 "X-Amz-Cf-Id":"b6fUKr6nyeph4sE1bHx8uDR1x7DA3y4OJ6uXVlCtMkCYbu5KKl7yoA=="}
```

SSL Certificate Store

- HTTP functions use the system default certificate store
  - /QIBM/USERDATA/ICSS/CERT/SERVER/DEFAULT.KDB
  - By default, does not exist
  - Digital Certificate Manager (DCM)
    - Creates the certificate store
    - Adds certificate to the store
  - Alternatively, create a KDB trust store from a JAVA trust store
    - See example in IBM Docs
      - SSL considerations section
      - https://www.ibm.com/docs/en/i/7.4?topic=programming-http-functions-overview

HTTP Functions

- What's new?
  - Headers option
    - Multiple headers can now be combined.

IBM i 7.5 TR 1
IBM i 7.4 TR 7

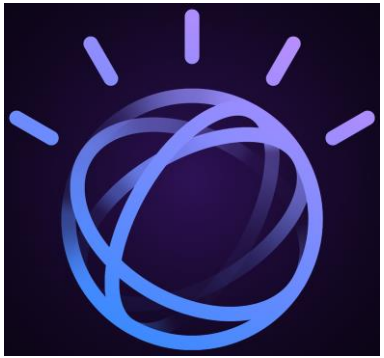
```
-- Previously needed to specify each header separately
VALUES QSYS2.HTTP_GET(
  'http://localhost:6050/get',
  '{"header":"user-Agent,MYAPP","header":"Content-Type,text/json"}');
```

```
-- Now can combine multiple headers
VALUES QSYS2.HTTP_GET(
  'http://localhost:6050/get',
  '{"headers":{"user-Agent":"MYAPP","Content-Type":"text/json"}}');
```

Db2 for i with Watson

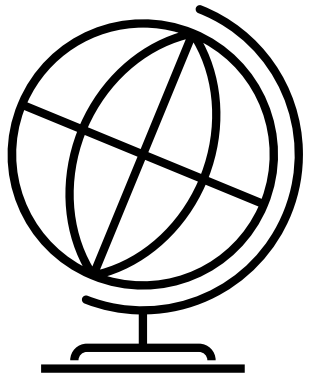
IBM Watson

- 2019 IBM Watson initiative
  - Embed Watson in many different IBM products
- Purpose
  - Bring the power of AI to many different IBM products
  - Standardize the adoption of AI solutions across IBM
- Common source
  - Eliminate redundancy
  - Foster community
  - Inspire innovation across IBM teams



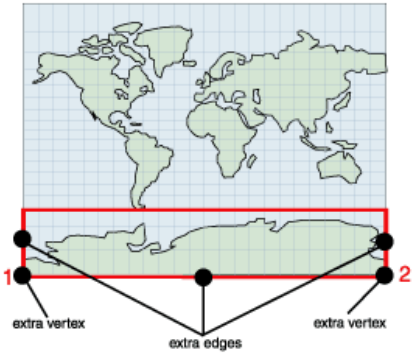
IBM Watson

- Watson spatio-temporal
  - Generate and analyze geospatial data
    - Discover insights about physical locations
  - Currently used by other IBM products
    - Maximo spatial
    - Watson studio
    - And others…



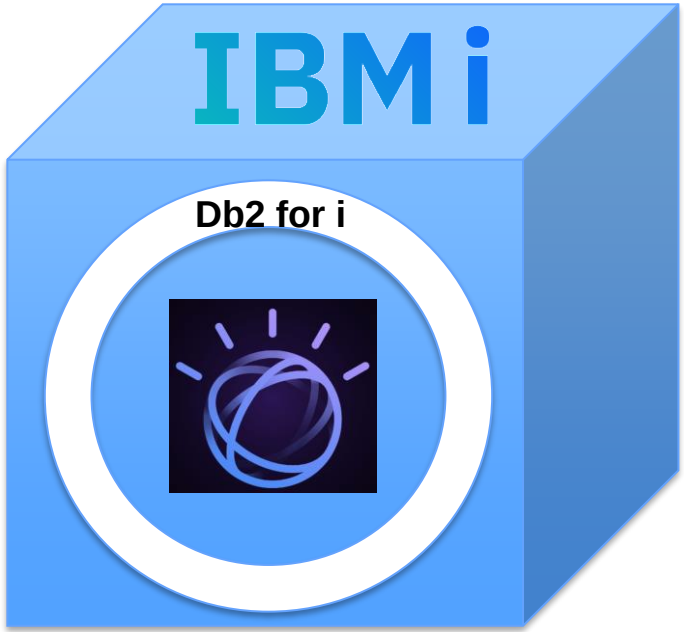
IBM Watson – Differentiator

- Coordinate system that describes a round, continuous, closed surface
- The earth is a globe with no edges or seams
  - Uses an ellipsoidal model of the Earth's surface
  - Geographic Coordinate System using WGS_1984
- Calculations are accurate and precise
- A flat-map projection requires the use of a projected geospatial reference system
  - No one projection can solve all scenarios
  - Not a problem with a round earth representation
    - Projection-less computations



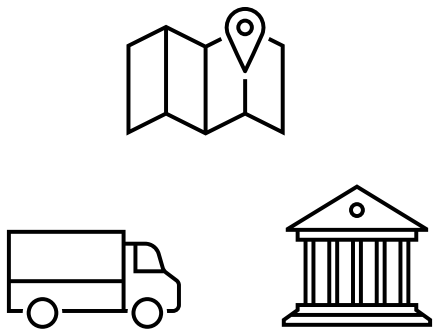
IBM Watson

- IBM Watson is integrated into Db2 for i
  - No web service calls, no reaching outside
  - No data moving off IBM i
  - No extra charge
- Uses SQL
  - QSYS2 user defined types
  - QSYS2 user defined functions
  - QSYS2 catalogs
- Db2 for i with Watson



What is Db2 Geospatial Analytics?

- Generate, store, and analyze geospatial information
  - You already have information about geographic features
    - Store locations, homes, shipping routes, business service areas
  - Understand the relationships between different geographies
    - Trucking management, advertising, tourism, flood assessment
  - Make business decision based on geospatial information
    - Where should I open a new franchise?



What is Db2 Geospatial Analytics?

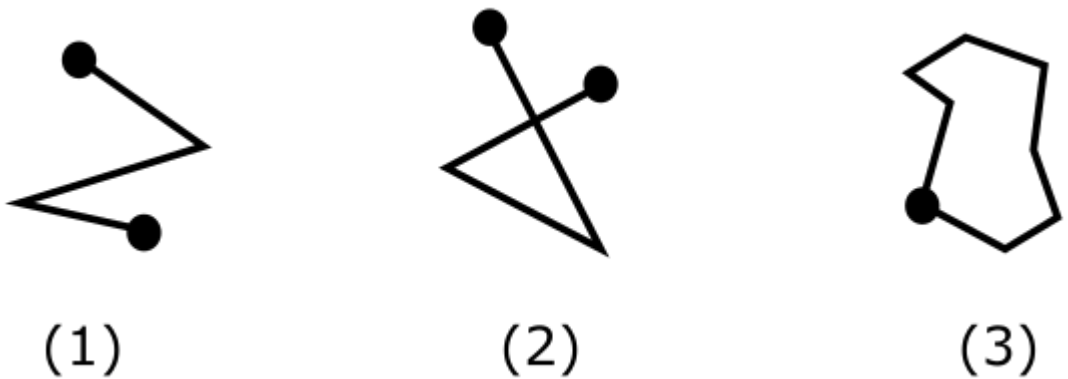
- Geospatial data
  - Coordinates that identify a location
    - x and y
    - Longitude and latitude
  - Geospatial data item
    - Single coordinate that identifies a location, like a store
    - Several coordinates that define a path, like a river
    - Coordinates that define a boundary of an area, like a flood plain

Types of geometries

- Point
  - A single point
  - Examples: House, store, city

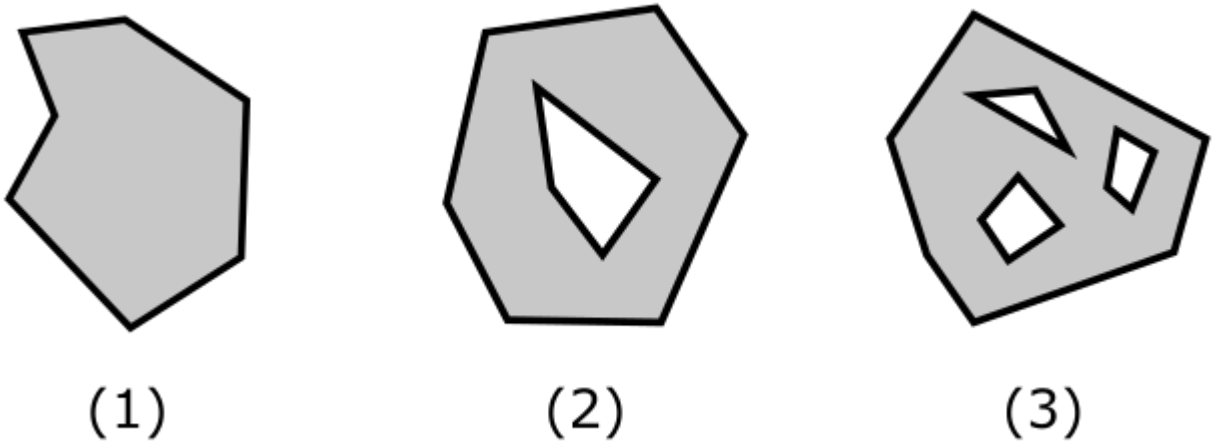
Types of geometries

- LINESTRING
  - A line between two or more points
  - Examples: streets, canals, and pipelines

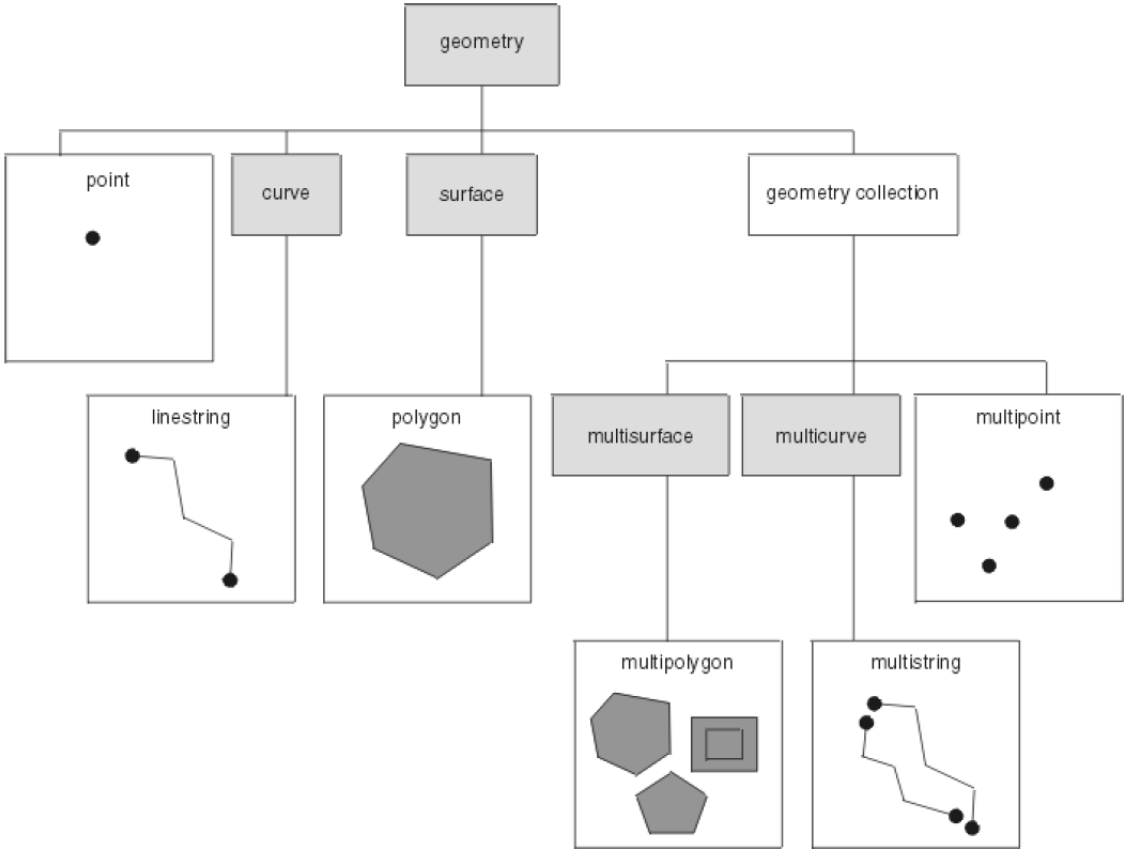


Types of geometries

- Polygons
  - A polygon or surface within a polygon
  - Examples: districts, forests, and wildlife habitats

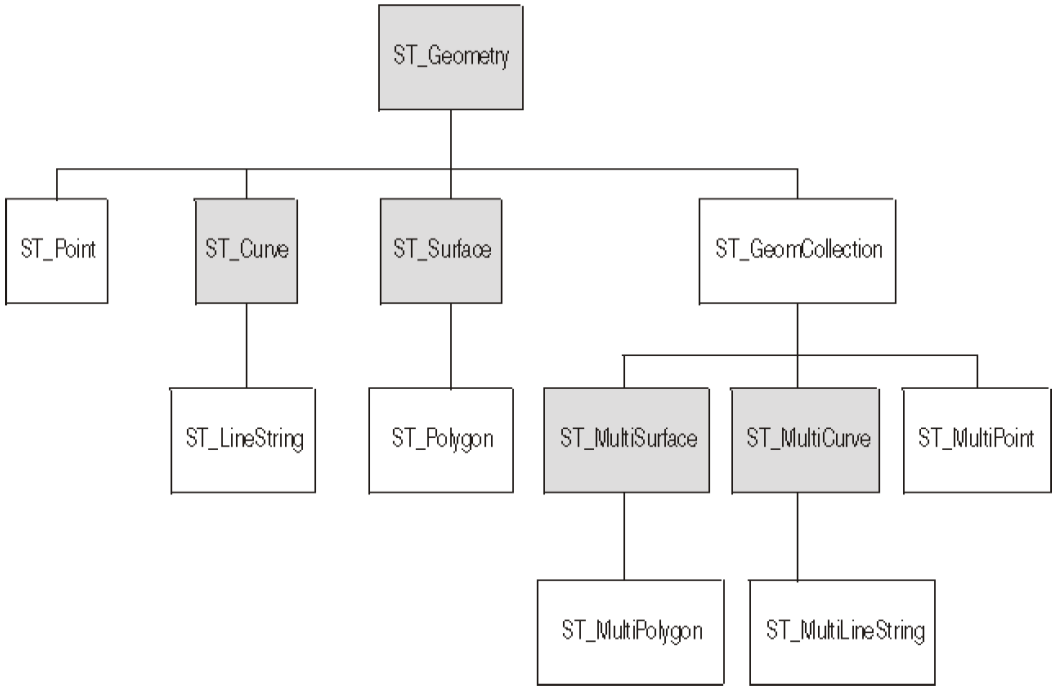


New data types



New data types

- User defined strong type
  - Based on a BLOB
- In QSYS2
- Hierarchy of types
  - All types are a ST_GEOMETRY
- Use the new constructor functions to generate data stored in a geospatial data types



Creating a table with a geospatial column

```
CREATE TABLE customer
  (id INT,
   lastname    VARCHAR(30),
   firstname   VARCHAR(30),
   address     VARCHAR(100),
   city        VARCHAR(50),
   postal_code VARCHAR(5),
   state       CHAR(2),
   location    QSYS2.ST_POINT);
```

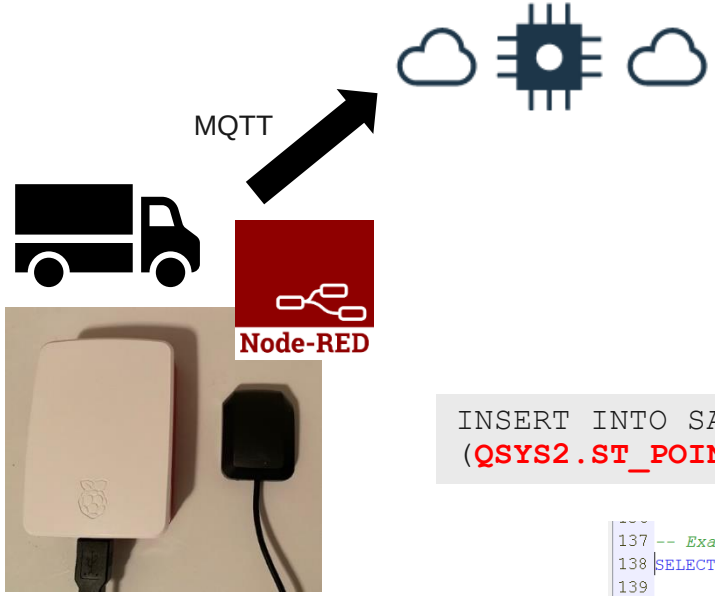
ID	LASTNAME	FIRSTNAME	ADDRESS	CITY	POSTAL_CODE	STATE	LOCATION
101	Kriner	Endela	9 Concourt Circle	San Jose	95141	CA	<Spatial Data BLOB>

Generating geospatial data

- Constructor scalar functions
  - Use coordinates to generate geospatial data
  - Scalar functions, located in QSYS2
  - Return a geospatial distinct type
    - ST_Point, ST_Linestring, ST_Polygon, ST_Multipoint, ST_Multilinestring, ST_Multipolygon, ST_Geomcollection
  - If you don't know what type, use the default ST_GEOMETRY

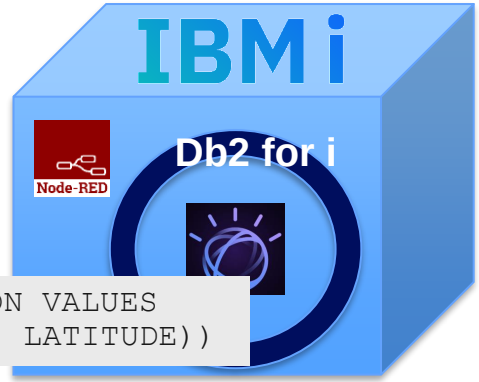
```
VALUES QSYS2.ST_Point('POINT (10 20)');
VALUES QSYS2.ST_Linestring('LINESTRING (10 10, 20 20, 21 30)');
VALUES QSYS2.ST_Polygon('POLYGON ((0 0, 0 40, 40 0, 0 0))');
VALUES QSYS2.ST_Geometry('GEOMETRYCOLLECTION(POINT(10 10),
                          LINESTRING(15 15, 20 20))');
```

Populating geospatial columns

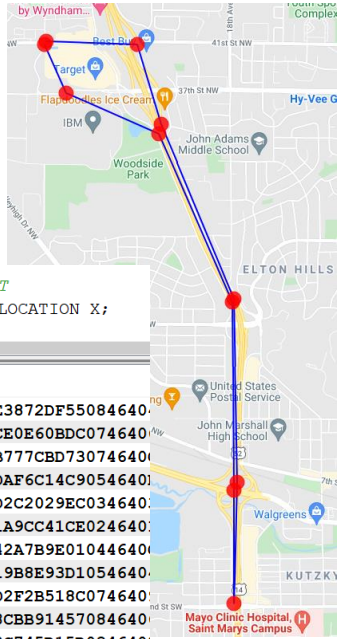


Raspberry Pi &
External GPS device

API call returns JSON Data:
{“Longitude”: -92.508, “Latitude”: 44.065}




```
INSERT INTO SARAHS_LOCATION VALUES
(QSYS2.ST_POINT(LONGITUDE, LATITUDE))
```




```
137 -- Example of location data from a GPS device sending data through Watson IOT
138 SELECT ST_ASTEXT(MY_LOCATION), MY_LOCATION AS LOCATION FROM GEOMETRY.SARAHS_LOCATION X;
139
```

00001	LOCATION
POINT (-92.50885799999999 44.065121)	01000000E6100000040000000100000004E3872DF55084640
POINT (-92.506497 44.061403)	01000000E6100000040000000100000006CE0E60BDC074640
POINT (-92.496562 44.05822)	01000000E6100000040000000100000CB777CBD730746400
POINT (-92.48862 44.045199)	01000000E6100000040000000100000BDAF6C14C9054640
POINT (-92.48845899999999 44.030645)	01000000E6100000040000000100000003D2C2029EC034640
POINT (-92.488486 44.021919)	01000000E6100000040000000100000011A9CC41CE024640
POINT (-92.488118 44.031299)	01000000E6100000040000000100000C42A7B9E01044640
POINT (-92.48848699999999 44.045457999999996)	01000000E6100000040000000100000419B8E93D1054640
POINT (-92.496297 44.058969999999995)	01000000E6100000040000000100000D2F2B518C074640
POINT (-92.49891699999999 44.065158)	01000000E6100000040000000100000068CBB91457084640
POINT (-92.50865499999999 44.065363)	01000000E6100000040000000100000023C745D15D084640

Populating geospatial columns

```
INSERT INTO SARAHS_LOCATION VALUES
  (QSYS2.ST_Point('POINT(-92.506497 44.061403)'))
```

```
137 -- Example of location data from a GPS device sending data through Watson IOT
138 SELECT ST_ASTEXT(MY_LOCATION), MY_LOCATION AS LOCATION FROM GEOMETRY.SARAHS_LOCATION X;
139
```

00001	LOCATION
POINT (-92.50885799999999 44.065121)	01000000E6100000400000001000000 4E3872DF55084640
POINT (-92.506497 44.061403)	01000000E6100000400000001000000 6CE0E60BDC074640
POINT (-92.496562 44.05822)	01000000E6100000400000001000000 CB777CBD73074640
POINT (-92.48862 44.045199)	01000000E6100000400000001000000 BDAF6C14C9054640
POINT (-92.48845899999999 44.030645)	01000000E6100000400000001000000 3D2C2029EC034640
POINT (-92.488486 44.021919)	01000000E6100000400000001000000 11A9CC41CE024640
POINT (-92.488118 44.031299)	01000000E6100000400000001000000 C42A7B9E01044640
POINT (-92.48848699999999 44.045457999999996)	01000000E6100000400000001000000 419B8E93D1054640
POINT (-92.496297 44.058969999999995)	01000000E6100000400000001000000 9D2F2B518C074640
POINT (-92.49891699999999 44.065158)	01000000E6100000400000001000000 68CBB91457084640
POINT (-92.50865499999999 44.065363)	01000000E6100000400000001000000 23C745D15D084640

New Geospatial Functions

Constructor functions
- ST_Geometry
- ST_Point
- ST_LineString
- ST_Polygon
- ST_GeomCollection
- ST_MultiPoint
- ST_MultiLinestring
- ST_MultiPolygon
- ST_WKTToSQL
- ST_WKBToSQL

Comparing Geometries
- ST_Contains
- ST_Covers
- ST_Crosses
- ST_Difference
- ST_Disjoint
- ST_Distance
- ST_Equals
- ST_Intersects
- ST_Overlaps
- ST_Touches
- ST_Within

Geometric Properties
- ST_Area
- ST_GeometryType
- ST_IsSimple
- ST_IsValid
- ST_MaxX
- ST_MaxY
- ST_MinX
- ST_MinY
- ST_NumPoints
- ST_SrsID
- ST_SrsName

New Geospatial Functions

Converting Geometries
- ST_AsText
- ST_AsBinary
- ST_ToPoint
- ST_ToLineString
- ST_ToPolygon
- ST_ToMultiPoint
- ST_ToMultiLine
- ST_ToMultiPolygon

Construct a new geometry
- ST_Buffer
- ST_Difference
- ST_Intersection
- ST_SymDifference
- ST_Union

Hash a geometry
- ST_FuzzyGeohashCover
- ST_ FuzzyGeohashCoverExtend
- ST_Geohash
- ST_GeohashCover
- ST_GeohashCoverExtend

Example

- Use ST_DISTANCE to find the distance between two points

```
-- Distance from IBM Rochester to Minneapolis International Airport
-- in kilometers

SELECT QSYS2.ST_Distance(QSYS2.ST_Point('point(-92.5055 44.0580)'),
                         QSYS2.ST_Point('point(-93.2223 44.8848)'))/1000
       AS DISTANCE
 FROM SYSIBM.SYSDUMMY1;
```

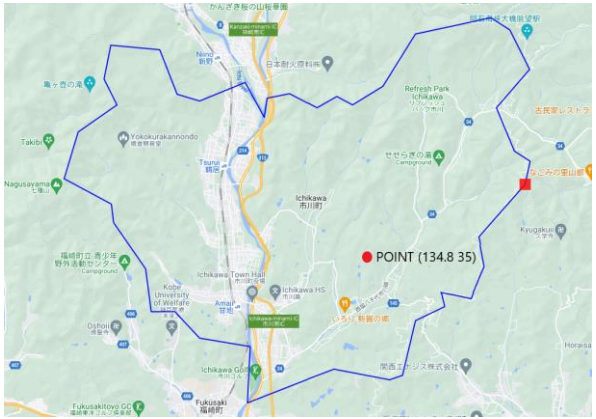
DISTANCE
108.22750834833546

Examples

```
-- The JAPAN_DISTRICTS table contains polygons in a column called
-- 'GEOMETRY' that represent different geographic districts in Japan.
-- Determine which district contains the location (134.8, 35)

SELECT objectid, QSYS2.ST_AsText(geometry) AS district, geometry
  FROM japan_districts
  WHERE QSYS2.ST_Contains(geometry, QSYS2.ST_Point('POINT (134.8 35)'));
```

OBJECTID	DISTRICT	GEOMETRY
1403	POLYGON ((134.84668 35.018477, 134.848207 35.024293, 134.8433729...	01000000E61000001...



Example – Storing geospatial data

```
-- Create three tables.
-- Table with the geometries of all the states in the US
-- Table with the center of 10,000 cities in the US
-- Paths of 10,000 historical tornados across the US

CREATE TABLE GEOTEST.US_STATES (STATE_ID CHAR(2) PRIMARY KEY,
                                STATE_FULL_NAME VARCHAR(50),
                                STATE_GEO QSYS2.ST_POLYGON);

CREATE TABLE GEOTEST.TORNADO_PATHS (TORNADO_ID CHAR(10) PRIMARY KEY,
                                    STATE CHAR(2),
                                    TORNADO_GEO QSYS2.ST_LINESTRING);

CREATE TABLE GEOTEST.US_CITIES (CITY_ID VARCHAR(10) PRIMARY KEY,
                                CITY_NAME VARCHAR(50),
                                STATE CHAR(2),
                                CITY_GEO QSYS2.ST_POINT);
```

Example – Populating data

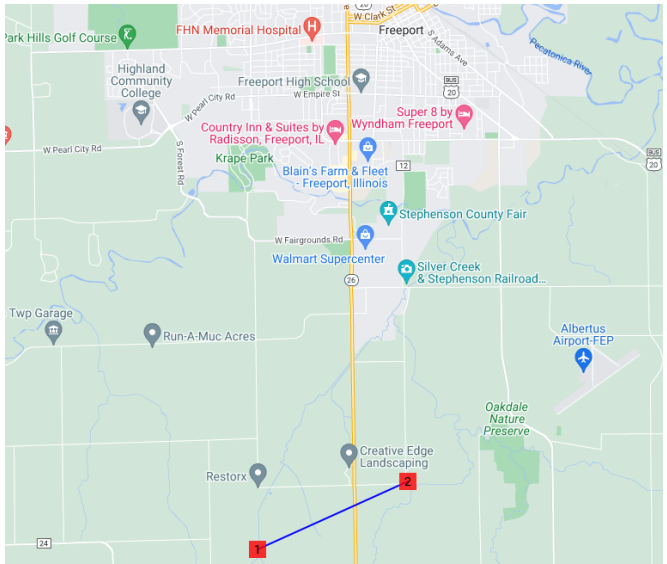
```
-- Populate the tables
-- Example - Populate table with 10,000 Cities
INSERT INTO GEOTEST.US_CITIES
  SELECT CITY_ID, NAME, STATE, QSYS2.ST_POINT(LONGITUDE, LATITUDE) FROM
    JSON_TABLE(QSYS2.HTTP_GET('https://public.opendatasoft.com/api/
                               records/1.0/search/?dataset=geonames-all-
                               cities-with-a-population-1000&q=&rows
                               =10000&sort=name&facet=feature_code&facet
                               =cou_name_en&facet=timezone&refine.
                               cou_name_en=United+States',
                              '{"header":"User-Agent,Sarah",
                                "sslCertificateStoreFile":
                                "/home/javaTrustStore/fromJava.KDB"}'),
               'lax $.records[*].fields'
               COLUMNS(CITY_ID   VARCHAR(10)  PATH '$.geoname_id',
                       NAME      VARCHAR(100) PATH '$.ascii_name',
                       STATE     VARCHAR(100) PATH '$.admin1_code',
                       LONGITUDE DOUBLE       PATH '$.coordinates[1]',
                       LATITUDE  DOUBLE       PATH '$.coordinates[0]'));
```

Example - Simple query

- Use ST_DISTANCE to find distance between a point and a linestring

```
-- How many tornados within 10 km of Freeport IL
SELECT TORNADO_ID, ST_ASTEXT(TORNADO_GEO)
  FROM GEOTEST.US_CITIES C, GEOTEST.TORNADO_PATHS P
  WHERE CITY_NAME = 'FREEPORT' AND C.STATE = 'IL' AND P.STATE = 'IL'
    AND QSYS2.ST_DISTANCE(CITY_GEO, TORNADO_GEO) < 10000;
```

TORNADO_ID	00002
44626	LINESTRING (-89.65000599999999 42.220008, -89.62000599999999 42.230008)



Example – Deeper analysis

- Data in the table doesn't tell the whole story.
- The state column only tells where the tornado started, but not where it ended.
- Tornado could have started in another state and crossed into Minnesota

```
-- What is the path of tornados that started in Minnesota
SELECT TORNADO_ID, STATE, QSYS2.ST_ASTEXT(TORNADO_GEO) AS PATH
  FROM GEOTEST.TORNADO_PATHS P
  WHERE STATE = 'MN';
```

TORNADO_ID	STATE	PATH
366	MN	LINESTRING (-94.77000799999999 43.570008, -94.480008 43.600007999999995)
959	MN	LINESTRING (-94.680008 43.750008, -94.47000799999999 43.980008)
3380	MN	LINESTRING (-93.570008 43.670007999999996, -93.56988 43.670096)
3796	MN	LINESTRING (-91.870008 47.180008, -91.86987099999999 47.180094)
5286	MN	LINESTRING (-96.200009 47.030007999999995, -96.199878 47.030097999999995)
6283	MN	LINESTRING (-92.980008 45.680008, -92.97987499999999 45.680095)
7858	MN	LINESTRING (-95.22000899999999 46.520008, -94.370008 46.530007999999995)
7871	MN	LINESTRING (-94.50000899999999 48.150008, -94.499873 48.150096)
10035	MN	LINESTRING (-93.650008 44.420007999999996, -93.649879 44.420096)

Example – Deeper analysis

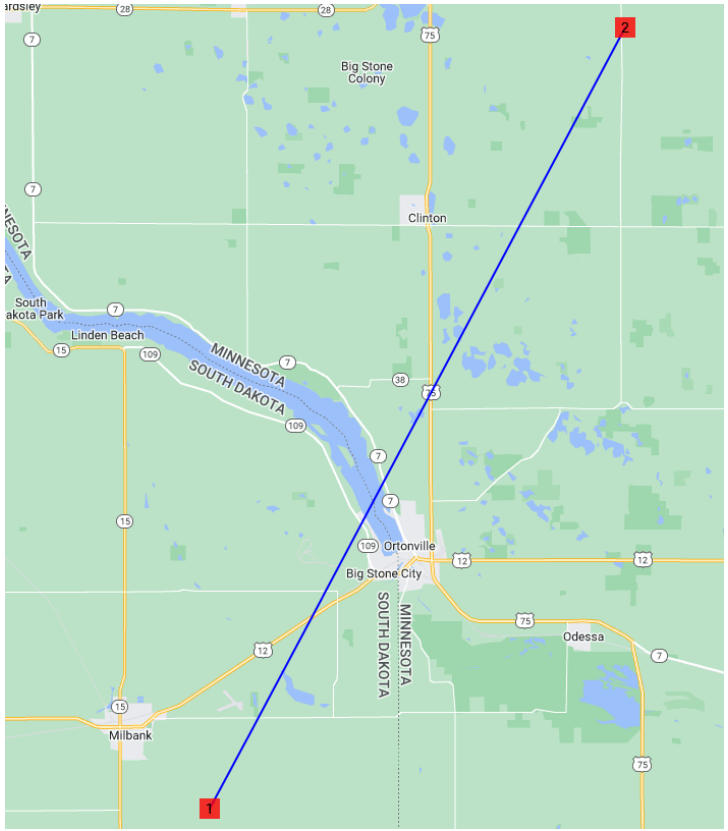
- Use ST_INTERSECTS to find tornados that started in another state but intersected with Minnesota

```
-- 4 tornados traveled into Minnesota from another state
SELECT TORNADO_ID,
       T.STATE AS PATH_START_STATE,
       S.STATE_FULL_NAME AS INTERSECT_STATE,
       QSYS2.ST_ASTEXT(TORNADO_GEO) AS TORNADO_GEO
  FROM GEOTEST.US_STATES S, GEOTEST.TORNADO_PATHS T
  WHERE S.STATE_ID = 'MN'
    AND T.STATE <> 'MN'
    AND ST_INTERSECTS(STATE_GEO, TORNADO_GEO) = 1;
```

TORNADO_ID	PATH_START_STATE	INTERSECT_STATE	TORNADO_GEO
56854	ND	Minnesota	LINESTRING (-97.200009 48.540008, -97.071709 48.505407999999996)
3358	ND	Minnesota	LINESTRING (-97.350009 46.900008, -96.230009 46.870008)
35802	SD	Minnesota	LINESTRING (-96.58000899999999 45.180008, -96.30000899999999 45.550008)
10710	SD	Minnesota	LINESTRING (-96.47000899999999 44.400008, -96.13000799999999 44.370008)

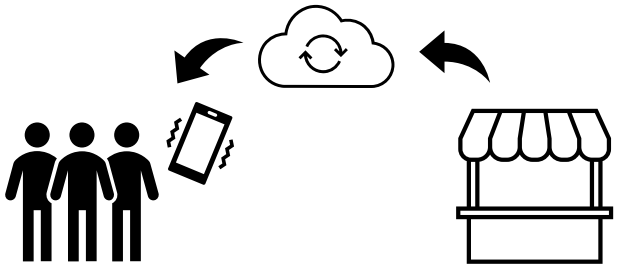
Example – Deeper analysis

- Use ST_INTERSECTS to find tornados that started in another state by intersected with Minnesota
- PATH_START_STATE = ‘SD’
- INTERSECT_STATE = ‘Minnesota’
- LINESTRING (-96.58000899999999 45.180008,
  -96.30000899999999 45.550008)
- Started in South Dakota, traveled into Minnesota



Taking it to the next level

- Integrating AI features
  - Map matching
    - Matching coordinates to the real world
      - Know where all your buses are and whether they'll reach their next stop on schedule
    - https://www.ibm.com/cloud/blog/real-time-map-matching-streaming-analytics-service
  - Hangout detection
    - Entity is "hanging out" if they have been in the same area for a period of time
      - Marketing opportunities – Offering promotions in real time
      - Transportation – Has a vehicle made the correct stops
    - https://www.ibm.com/blogs/cloud-archive/2016/05/real-time-hangout-detection/



END PART 1

Database Engineer

SQL Error Logging Facility (SELF)

- Common Problem: How do I find, understand, and fix SQL errors and warnings?
  - Trying to identify a problem when it happens is difficult
  - Performance monitors can be helpful, but come with overhead

- Solution: SQL Error Logging Facility
  - Captures detail about user specified SQL errors and warnings

SQL Error Logging Facility (SELF)

- How does it work?
  - User identify specific SQLCODE values
    - SYSIBMADM.SELFCODES built-in global variable
  - SELF runs when an SQL statement returns the SQLCODE
  - SELF collects point-of-failure information
    - QSYS2.SQL_ERROR_LOG view
- No performance impact on:
  - Statements that run successfully
  - Statements that return a SQL error or warning not in SELFCODES

SQL Error Logging Facility (SELF)

- What is recorded
  - Program Name
  - SQL statement text and operation
  - Reason Code
  - Program information
  - Job name
  - User information
  - Call stack
  - and more …

SQL Error Logging Facility (SELF)

- Easy to use
  - By default, `SYSIBMADM.SELFCODES` is set to NULL
    - Nothing is logged
  - Enabled in a job or system wide

```
-- Set the SELFCODES for all jobs
CREATE OR REPLACE VARIABLE SYSIBMADM.SELFCODES VARCHAR(256) DEFAULT '-913, -206';

-- Set the SELFCODES within a specific job.
SET SYSIBMADM.SELFCODES = SYSIBMADM.VALIDATE_SELF ('-514, -204, -501, -199');

VALUES SYSIBMADM.SELFCODES;
```

```
00001
-514, -204, -501, -199
```

SQL Error Logging Facility (SELF)

- Review the SELF detail

```
-- Query the SELF view
SELECT * FROM QSYS2.SQL_ERROR_LOG;
```

LOGGED_SQLCODE	LOGGED_SQLSTATE	NUMBER_OCCURRENCES	STATEMENT_OPERATION_DETAIL	STATEMENT_TEXT	INITIAL_USER_STACK
-204	42704	1	CALL	CALL QSYS2.QCMDEXC(?)	
-601	42710	2	CREATE TABLE	-- force a SQL0601 error CREATE TABLE JGLIB.F0101 (COL1 INT...	
-204	42704	1	CREATE PROCEDURE	---cursor CREATE or REPLACE PROCEDURE cur_proc1 LANGUAGE SQ...	
-601	42710	3	CREATE TABLE	CREATE TABLE JGLIB.F0101 (COL1 INTEGER)	
-672	55035	1	DROP	drop table t1	[{"ORD":13,"TYPE":'
-104	42601	1	EXECUTE IMMEDIATE	abc	
-204	42704	1	INSERT	insert into table1 values(?,?),(?,?),(?,?)	
-672	55035	2	DROP	---to try drop table qgpl.t1	
-420	22018	3	UNKNOWN	UPDATE TABLE1 SET C1 = 'abc' , C2 = 'xx1' WHERE CURRENT OF ...	[{"ORD":19,"TYPE":'
-420	22018	1	CALL	call cur_proc1a()	
-811	21000	3	SET VARIABLE	SET : H : H = ((SELECT NUMBER_OCCURRENCES FROM QSYS2 . SQ...	[{"ORD":40,"TYPE":'
-811	21000	2	OPEN	values selffun1()	
-204	42704	1	CALL	CALL QSYS2.QCMDEXC('RUNSQL SQL(''drop table tabc'')')	[{"ORD":13,"TYPE":'
-104	42601	1	EXECUTE IMMEDIATE	create table t1	

https://www.ibm.com/docs/en/i/7.5?topic=tools-sql-error-logging-facility-self

Reorganize Physical File Member

- What is it?
  - CL command
  - Remove fragmentation in a physical file’s member
  - Optionally reorganize the member
- Why use it?
  - I/O efficiency
  - Reduce space consumption

Reorganize Physical File Member

- ALWCANCEL *YES previous behavior

ALWCANCEL *NO	ALWCANCEL *YES
“Classic” style of reorganize	“New” style of reorganize
Makes a copy of the records	**Delete and insert active records**
Exclusive lock required	No exclusive lock required
All or nothing	Incremental progress is possible
Requires an outage for the file	No outage for the file
	Can be long running
	Journaling required
	Active records appear to be missing

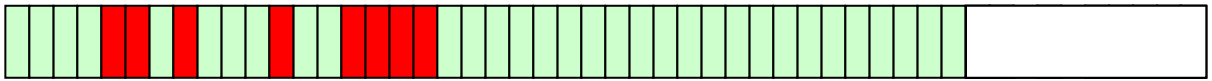
Reorganize Physical File Member

- ALWCANCEL *YES previous behavior

ALWCANCEL *NO	ALWCANCEL *YES
“Classic” style of reorganize	“New” style of reorganize
Makes a copy of the records	**Move active records**
Exclusive lock required	No exclusive lock required
All or nothing	Incremental progress is possible
Requires an outage for the file	No outage for the file
	Can be long running
	Journaling required
	~~**Active records appear to be missing**~~

Reorganize Physical File Member

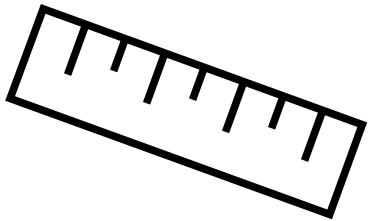
- Example
  - KEYFILE(*RPLDLTRCD)
    - Active records from end of the file moved to where deleted records are at the start of the file
    - 8 rows moved in the example



Example, 10 deleted rows (red) 40 active rows

Varying-length data

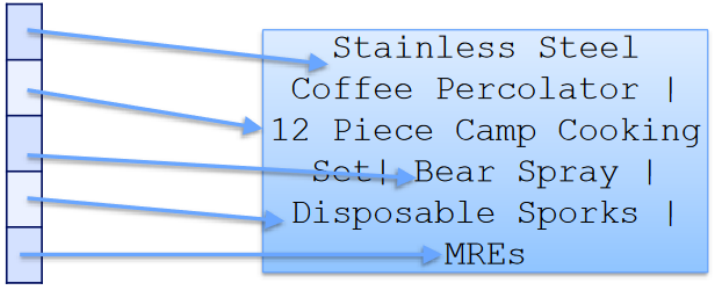
- Varying-length data
  - VARCHAR, BLOB, etc
  - Stored
    - Inline in fixed-length space
      - Read/write access more efficient
    - Separately in an overflow space
      - Use disk space more efficiently
- ALLOCATE attribute
  - Where is the data stored?
    - Inline
      - Length is less than or equal to ALLOCATE attribute
    - Overflow space
      - Length is greater than ALLOCATE attribute



Varying-length data

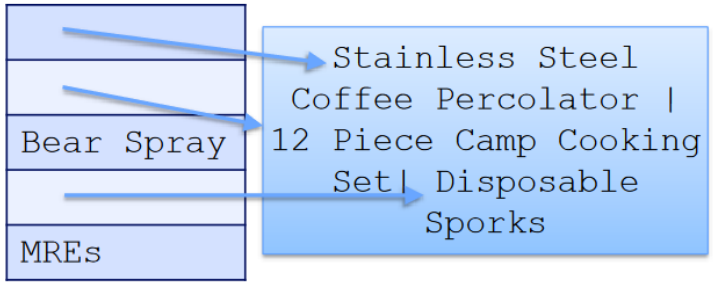
ITEM_DESCRIPTION – VARCHAR(50)	Length
Stainless Steel Coffee Percolator	33
12 Piece Camp Cooking Set	25
Bear Spray	10
Disposable Sporks	17
MREs	4

ALLOCATE(0)



Requires 33+25+10+17+4 = 89 bytes

ALLOCATE(10)



Requires 10 x 5 + (33+25+17) = 125 bytes

ALLOCATE(33)

Stainless Steel Coffee Percolator
12 Piece Camp Cooking Set
Bear Spray
Disposable Sporks
MREs

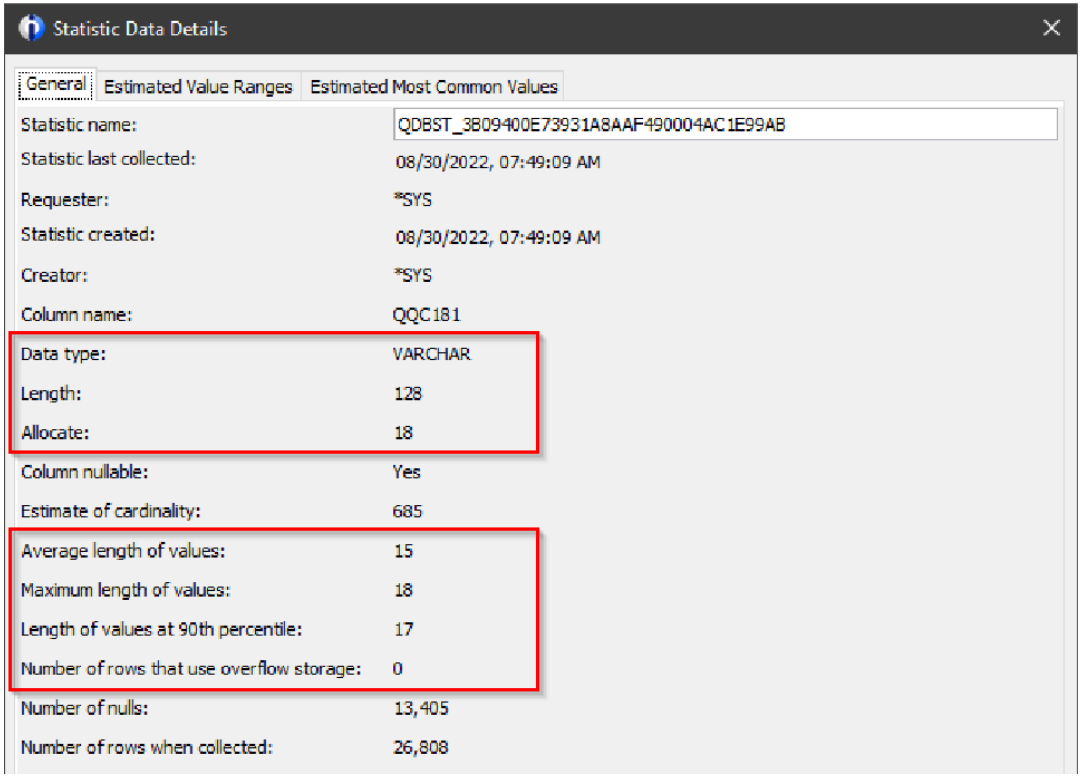
Requires 33 X 5 = 165 bytes

Column statistics

- Database statistics manager
  - Analyzes data
  - Helps the optimizer choose efficient query plans
- Column used by a query automatically analyzed
  - WHERE clause or JOIN conditions
  - Unless automatic collection is disabled
- Current analysis
  - Number of nulls
  - Number of distinct values
  - Most frequent values

Column statistics

- New analysis for varying-length columns
  - Average data length
  - Maximum data length
  - 90th percentile length
  - Count of overflow rows
- Access
  - Schema tool in ACS
  - QdbstListStatistics() API
  - QSYS2.SYSCOLUMNSTAT SQL view



Column statistics and varying-length fields

- Access via the QSYS2.SYSCOLUMNSTAT SQL view

```
-- Find tables where at least 10% of the rows use the overflow storage
SELECT TABLE_NAME, COLUMN_NAME, MAXIMUM_COLUMN_LENGTH,
       LENGTH_AT_90TH_PERCENTILE, OVERFLOW_ROWS, CURRENT_ROWS
 FROM QSYS2.SYSCOLUMNSTAT
 WHERE TABLE_SCHEMA = 'NQEDBMON'
   AND OVERFLOW_ROWS > .10 * CURRENT_ROWS
 ORDER BY OVERFLOW_ROWS DESC;
```

TABLE_NAME	COLUMN_NAME	MAXIMUM_COLUMN_LENGTH	LENGTH_AT_90TH_PERCENTILE	OVERFLOW_ROWS	CURRENT_ROWS
SRVRPROC6	QQ1000	1000	231	29079	32588
SRVRPROC7	QQ1000	1000	231	28721	30694
MA42588	QQ1000	1000	111	18212	126629
NIST	QQ1000	486	121	11749	26006
M84JFMED02	QQ1000	1000	70	11246	63677
MA40359	QQ1000	998	141	10663	37299

Column statistics and varying-length fields

- Column statistics can help you balance performance and storage demands by guiding your use of variable length columns.
- Overflow row count per column is only available from column statistics

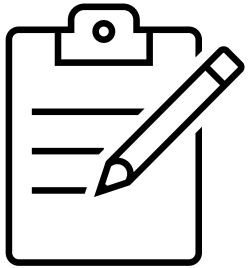
Query Supervisor and other query enhancements

Query Supervisor

- Monitor queries
  - Detect if a query exceeds resource consumption threshold
    - Both SQL and native queries
    - You define the threshold
  - When threshold is met or exceeded
    - User exit program is called
- You can now
  - Get notified when a query exceeds its resource threshold
    - Detect runaway queries
    - Take action by terminating queries
  - Capture and log details about a query
    - Help with understanding why a query ran different than normal
    - Understand temporary storage usage

Query Supervisor

- Thresholds you can define and monitor for:
  - CPU time
    - Total processing time
  - Elapsed time
    - Total clock time
  - Temporary storage
    - Amount of storage the query uses
  - Total I/O count



Query Supervisor

- When the threshold is met
  - Take action!
    - Exit programs registered with the Query Supervisor Exit Point are called
    - Query is paused until exit program completes
    - Information about the query is passed to the exit program
      - Log information
      - Send a message
      - Retrieve information about the query and use it for analysis
      - Optionally terminate the query

https://www.ibm.com/docs/en/i/7.4?topic=tools-query-supervisor

Db2 Symmetric Multiprocessing (SMP)

- SMP (Option 26) is now available as a no-charge option
  - SMP is intended for long-running (OLAP) queries and batch workloads
  - SMP is typically not very helpful for OLTP (transactional) workloads
- Query optimizer tries to make each query run as fast as possible
  - New options to shape SMP performance
    - Minimum time before using SMP
    - Maximum system CPU used
  - For PARALLEL_DEGREE *OPTIMIZE only
  - Configured with QAQQINI

SMP :: PARALLEL_DEGREE *OPTIMIZE [percentage]

IBM i 7.5 TR 1

- Existing option with new meaning
  - **Control how much processing power the optimizer may consider during optimization**
    - Previously applied after optimization finished, but before execution
    - Provide with more accurate information to number of tasks
  - By default, the optimizer assumers it can use as many threads as cores on the system
    - Values < 100 reduce the number of cores considered available
    - Values > 100 are possible but rarely useful
    - Use *MAX when need to use all available processing power

SMP :: PARALLEL_MAX_SYSTEM_CPU percentage

IBM i 7.5 TR 1

- Protects systems from SMP “overload”
  - Dynamically adjust running queries to keep CPU usage below a safe threshold
  - If CPU usages exceeds 80%
    - Query tasks will be paused until CPU usage falls below 80%
    - Tasks will resume running when under threshold
    - There will always by at least one task running
    - Reducing the number running of tasks will slow, but not stop queries
  - The default value of 80% may be modified
    - QAQQINI parameter
      - PARALLEL_MAX_SYSTEM_CPU

SMP :: PARALLEL_MIN_TIME

IBM i 7.5 TR 1

- Reserve the use of SMP for queries that need it
  - Long running analytical queries
- Optimizer determines if a query needs SMP based on how long it runs
  - Optimizer start out as if PARALLEL_DEGREE was *NONE
  - Adaptive Query Processing detect queries that run longer than configured time
    - Query has not returned data
      - Query is re-optimized with SMP
      - New plan is swapped in
    - Query has returned data
      - Query will run to completion
      - Optimizer with re-optimize with SMP next time query is run
    - Decisions are remembered until IPL
- Default time is 60 seconds
  - Can be changed with PARALLEL_MIN_TIME

MTI_INFO

- Available in 7.3 and later
  - Detailed information about active Maintained Temporary Index (MTI)
  - Index advisor → What the optimizer recommended in the past
  - MTI_INFO → What is on the system NOW
  - Requires *JOBCTL special authority and QIBM_DB_SQLADM function usage
- 7.5 TR1
  - Now returns QRO hash and Plan ID
  - Returned as JSON

MTI_INFO

- Find queries that use a lot of temporary storage

```
-- Find large MTI
SELECT TABLE_SCHEMA, TABLE_NAME, MTI_SIZE, QRO_HASH_JSON, PLAN_IDENTIFIER_JSON
 FROM TABLE(QSYS2.MTI_INFO())
  WHERE TABLE_SCHEMA NOT LIKE 'Q%'
  ORDER BY MTI_SIZE DESC;
```

TABLE_SCHEMA	TABLE_NAME	MTI_SIZE	QRO_HASH_JSON	PLAN_IDENTIFIER_JSON
JDTESTINFO	SCSTA1	817816	{"QRO HASH LIST" : ["4401A287", "A6CBA901"] }	{"PLAN IDENTIFIER" : [10493, 10494, 16894, 18419, 24447] }
JDTESTINFO	SCHED1	449176	{"QRO HASH LIST" : ["BE33995", "326748F5", "9908C4...	{"PLAN IDENTIFIER" : [167, 3988, 3989, 4006, 5426, 5427, 542

New and Enhanced IBM i services

COMMAND_INFO

- Similar to
  - Display Command (DSPCMD)
  - Retrieve Command Information (QCDRCMDI) API

```
-- Display commands for which only a limited user can call

SELECT COMMAND_LIBRARY, COMMAND_NAME, TEXT_DESCRIPTION FROM QSYS2.COMMAND_INFO
  WHERE ALLOW_LIMITED_USER = 'YES';
```

COMMAND_LIBRARY	COMMAND_NAME	TEXT_DESCRIPTION
QIWS	STRPCO	Start PC Organizer
QSYS	DSPJOB	Display Job
QSYS	DSPJOBLOG	Display Job Log
QSYS	DSPMSG	Display Messages
QSYS	SIGNOFF	Sign Off
QSYS	SNDMSG	Send Message
QSYS	WRKMSG	Work with Messages

HARDWARE_RESOURCE_INFO

- Both a UDTF and a view
- SQL alternatives to
  - STRSST Hardware Service Manager interface
  - Retrieve Hardware Resource List (QGYRHRL, QgyRtvHdwRscList) API
  - Retrieve Hardware Resource Information (QGYRHRI, QgyRtvHdwRscInfo) API
- The UDTF includes resource category and detail query performance controls

```
-- Retrieve information for all Storage device resources
SELECT RESOURCE_CATEGORY, RESOURCE_NAME, STATUS, TEXT_DESCRIPTION
  FROM TABLE (QSYS2.HARDWARE_RESOURCE_INFO(RESOURCE_CATEGORY => 'STORAGE',
                                           DETAILED_INFO => 'NO'));
```

RESOURCE_CATEGORY	RESOURCE_NAME	STATUS	TEXT_DESCRIPTION
STORAGE	DMP043	OPERATIONAL	Disk Unit
STORAGE	CMB06	OPERATIONAL	Comm Processor
STORAGE	DC03	OPERATIONAL	Storage Controller
STORAGE	OPTVRT36	OPERATIONAL	Optical Storage Unit

SYSDISKSTAT

- SYSDISKSTAT enhanced:
  - The SYSDISKSTAT view contains information about spinning disk and solid-state drives (SSD)
  - HOST_WWPN

    A hexadecimal string representing the resource's host world wide port name.
  - REMOTE_WWPN

    A hexadecimal string representing the resource's remote world wide port name.

```
-- Display disk  status information
SELECT ASP_NUMBER, RESOURCE_NAME, HOST_WWPN, REMOTE_WWPN FROM QSYS2.SYSDISKSTAT;
```

CHANGE_DISK_PATHS

CHANGE_DISK_PATHS

- Enables or disables all paths matching the parameters provided
- This procedure is only supported for some external SCSI devices

```
-- Display disk  status information
CALL QSYS2.CHANGE_DISK_PATHS(OPERATION => 'ENABLE',
                             HOST_WWPN => 'c050760a3038001e');
```

Understanding system limits

- Automatic tracking of high consumers of important system resources
- Messages sent to QSYSOPR to alert for a subset of important system limits
- Full details here: ibm.biz/Db2foriAlerts

Limit ID	Limit Description	Maximum	Alerting Level	Alerting Cadence
15000	Maximum number of all rows in a partition	4,294,967,288	Greater than 90%	Once per day
15003	Maximum size of the data in a table partition	1,869,169,767,219		
15104	Maximum number of variable-length segments	65,533		
15400	Maximum *MAX4GB Index Size	4,294,967,296		
15401	Maximum *MAX1TB Index Size	1,869,166,411,776		
15403	Maximum Encoded Vector Index Size	2,199,023,255,552		
19002	Maximum number of spooled files in the system and basic user ASPs	2,610,000		

Understanding system limits

- Sent once per day
- Defaults to 90%, but can be customized using Global Variables in SYSIBMADM
- Reaching a system limit can cause outages
- Once an alert is received
  - Review and access the alert
  - Evaluate growth trends to determine how quickly action is needed
  - Investigate different methods to overcome the limits
  - Identify a solution
  - Implement the solution

PROCESS_SYSTEM_LIMITS_ALERTS

- System limits by default sent
  - When Collection Services recycles
  - Typically once per day, just past midnight
- Call PROCESS_SYSTEM_LIMITS_ALERTS to process alerts for the past 24 hours
  - Allows you to alert more frequently than once a day.

```
--Process system alerts for the last 24 hours

CALL QSYS2.PROCESS_SYSTEM_LIMITS_ALERTS();
```

ObjectConnect Services

- CHANGE_OBJECTCONNECT() procedure
  – Configures, Starts, or Stops the ObjectConnect over IP server

```
--Start the ObjectConnect server

CALL QSYS2.CHANGE_OBJECTCONNECT(STATE => 'START');
```

- OBJECTCONNECT_INFO view
  – Returns information about the ObjectConnect over IP server

```
--Review the Object Connect server information

SELECT * FROM QSYS2.OBJECTCONNECT_INFO;
```

New columns added to existing services

- History_Log_Info
  - FROM_JOB_NAME, FROM_JOB_USER, and FROM_JOB_NUMBER
- Asp_Info
  - ENCRYPTED_ASP + 7 Geographic mirror columns
- Output_Queue_Info
  - INTERNET_ADDRESS
- Active_Job_Info
  - Include WORKLOAD_GROUP when DETAIL_INFO = ‘WORK’
- System_status & System_Status_Info
  - 5 Job Summary columns
  - 9 Batch Summary columns

Audit Journal Helper Function

- QSYS/QAUDJRN used for security auditing
- Many new audit journal services
- Returns information from the audit journal specific to an individual entry type
  - AD (Auditing Change)
  - DS (Service Tools User ID and Attribute Changes)
  - IM (Intrusion Monitor)
  - PG (Primary Group Change)
  - SK (Sockets Connections)
  - SM (Systems Management Change)
  - ZR (Read of Object)
  - ZC (Change of Object)

Audit Journal Helper Function

- SYSTOOLS.AUDIT_JOURNAL_AD
  - Returns rows from the audit journal for the AD (Auditing Change) journal entries.

```
--Review the Object Connect server information

SELECT LIBRARY_NAME, OBJECT_NAME, OBJECT_TYPE ,
       ENTRY_TYPE_DETAIL, OBJECT_AUDIT, PREV_OBJECT_AUDIT
  FROM TABLE(SYSTOOLS.AUDIT_JOURNAL_AD());
```

LIBRARY_NAME	OBJECT_NAME	OBJECT_TYPE	ENTRY_TYPE_DETAIL	OBJECT_AUDIT	PREV_OBJECT_AUDIT
SAMTEST	T1	*FILE	CHGOBJAUD or CHGAUD command	*ALL	*NONE

CREATE_USER_INDEX

- Creates or replaces a user index (*USRIDX) object.
- Similar to Create User Index (QUSCRTUI) API.

```
CALL QSYS2.CREATE_USER_INDEX(USER_INDEX => 'USRINDX1',
                             USER_INDEX_LIBRARY => 'SAMTEST',
                             ENTRY_TYPE => 'VARIABLE',
                             KEY_LENGTH => 5,
                             MAXIMUM_ENTRY_LENGTH => 100,
                             PUBLIC_AUTHORITY => '*EXCLUDE');
```

CREATE_USER_SPACE

- Creates or replaces a user space (*USRSPC) object
- Similar the Create User Space (QUSCRTUS) API

```
CALL QSYS2.CREATE_USER_SPACE(USER_SPACE => 'USRSPC1',
                             USER_SPACE_LIBRARY => 'SAMTEST',
                             SIZE => 1000,
                             PUBLIC_AUTHORITY => '*EXCLUDE’);
```

CHANGE_USER_SPACE

- Creates or replaces a user index (*USRIDX) object.
- Similar to Create User Index (QUSCRTUI) API.

```
CALL QSYS2.CHANGE_USER_SPACE(USER_SPACE => 'USRSPC1',
                             USER_SPACE_LIBRARY => 'SAMTEST',
                             DATA => 'Overwrite with this new value',
                             START_POSITION => 500);
```

CHANGE_USER_SPACE_ATTRIBUTES

- Change the attributes of a user space (*USRSPC) object.
- Similar to Change User Space Attributes (QUSCUSAT) API.

```
CALL QSYS2.CHANGE_USER_SPACE_ATTRIBUTES(USER_SPACE => 'USRSPC1',
                                        USER_SPACE_LIBRARY => 'SAMTEST',
                                        SIZE => 200);
```

ASSOCIATE_JOURNAL_RECEIVER

- Associates a journal receivers with a journal if the journal receiver was originally associated with the journal.
- This was done previously through green screen only (WRKJRN OPTION 9)

```
-- Create two journal receivers and a journal
CL: CRTJRNRCV SAMTEST/RCV1;
CL: CRTJRNRCV SAMTEST/RCV2;
CL: CRTJRN SAMTEST/JRN1 SAMTEST/RCV1;
CL: CHGJRN JRN(SAMTEST/JRN1) JRNRCV(SAMTEST/RCV2);

-- Save, delete, and restore the journal. Receivers no longer associated with journal
CL: SAVOBJ OBJ(JRN1) LIB(SAMTEST) DEV(*SAVF) SAVF(SAMACKEN/SAVE1);
CL: DLTOBJ SAMTEST/JRN1 *JRN;
CL: RSTOBJ OBJ(JRN1) SAVLIB(SAMTEST) DEV(*SAVF) SAVF(SAMACKEN/SAVE1);

-- Associate journal receivers with journal
SELECT *
   FROM TABLE(QSYS2.ASSOCIATE_JOURNAL_RECEIVER('SAMTEST', 'JRN1', 'SAMTEST', 'RCV*'));
```

JOURNAL_LIBRARY	JOURNAL_NAME	RECEIVER_LIBRARY	RECEIVER_NAME	RESULT	RESULT_DETAIL	RETURN_CODE
SAMTEST	JRN1	SAMTEST	RCV1	SUCCESS	Journal receiver successfully associated with journal	0
SAMTEST	JRN1	SAMTEST	RCV1002	WARNING	Journal receiver already assocaited with journal	1
SAMTEST	JRN1	SAMTEST	RCV2	SUCCESS	Journal receiver successfully associated with journal	0

JOURNAL_RECEIVER_INFO

- A view that contains information about all journal receivers on the system.
- Similar to Retrieve Journal Receiver Information (QjoRtvJrnReceiverInformation) API

```
SELECT JOURNAL_RECEIVER_LIBRARY, JOURNAL_RECEIVER_NAME, JOURNAL_LIBRARY, JOURNAL_NAME,
       DETACH_TIMESTAMP
  FROM QSYS2.JOURNAL_RECEIVER_INFO
  WHERE JOURNAL_RECEIVER_LIBRARY = 'SAMTEST';
```

Journal Receiver Library JOURNAL_RECEIVER_LIBRARY	Journal Receiver Name JOURNAL_RECEIVER_NAME	Journal Library JOURNAL_LIBRARY	Journal Name JOURNAL_NAME	Detach Timestamp DETACH_TIMESTAMP
SAMTEST	RCV1	SAMTEST	JRN1	2022-10-18 08:59:55.000000
SAMTEST	RCV1002	SAMTEST	JRN1	<NULL>
SAMTEST	RCV2	SAMTEST	JRN1	2022-10-18 08:59:58.000000

REMOTE_JOURNAL_INFO

- Returns information about every remote journal defined for a local or remote journal
- Similar to Retrieve Journal Information (QjoRetrieveJournalInformation) API

```
SELECT SOURCE_JOURNAL_LIBRARY, SOURCE_JOURNAL, REMOTE_DATABASE_NAME, REMOTE_JOURNAL
  FROM QSYS2.REMOTE_JOURNAL_INFO
  WHERE REMOTE_DATABASE_NAME = 'UT23P37';
```

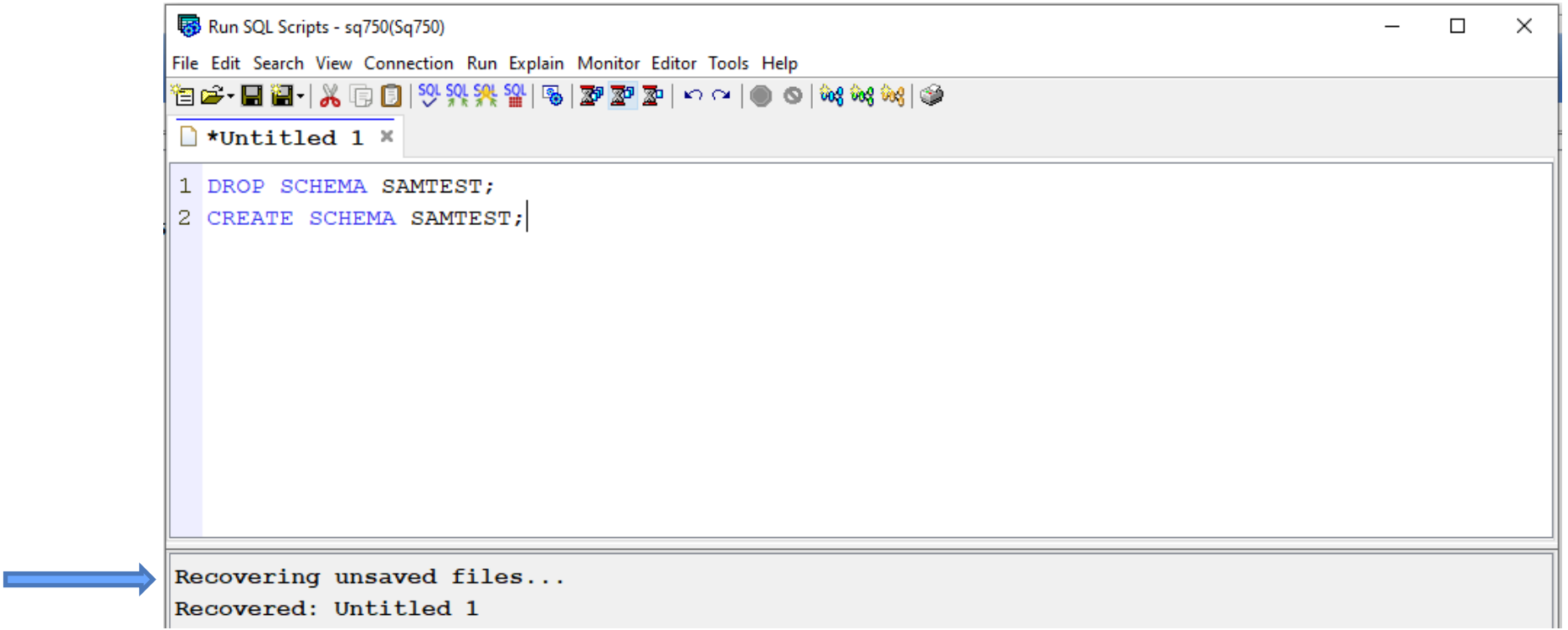
Source Journal Library SOURCE_JOURNAL_LIBRARY	Source Journal SOURCE_JOURNAL	Remote Database Name REMOTE_DATABASE_NAME	Remote Journal REMOTE_JOURNAL
SAMTEST	JRN1	UT23P37	JRN1

IBM i Access Client Solutions (ACS)

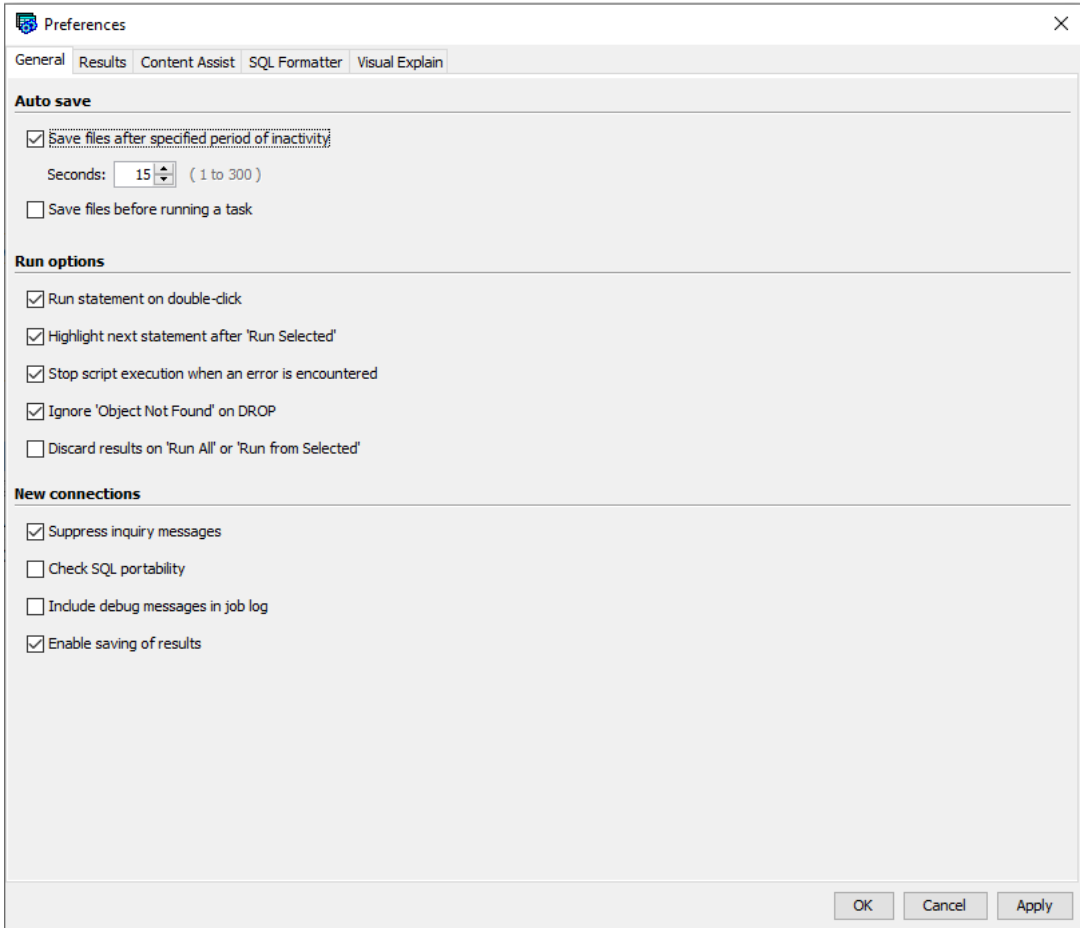
Run SQL scripts Protection

- Add crash recovery support
- The contents of unsaved editor tabs will be continuously mirrored to temporary files. In normal processing, these files will be cleaned up when the editor tabs are closed
- **In the event of a crash, the temporary files will remain**
- When Run SQL Scripts is restarted, the temporary files will appear under tabs in the new session

Run SQL scripts Protection

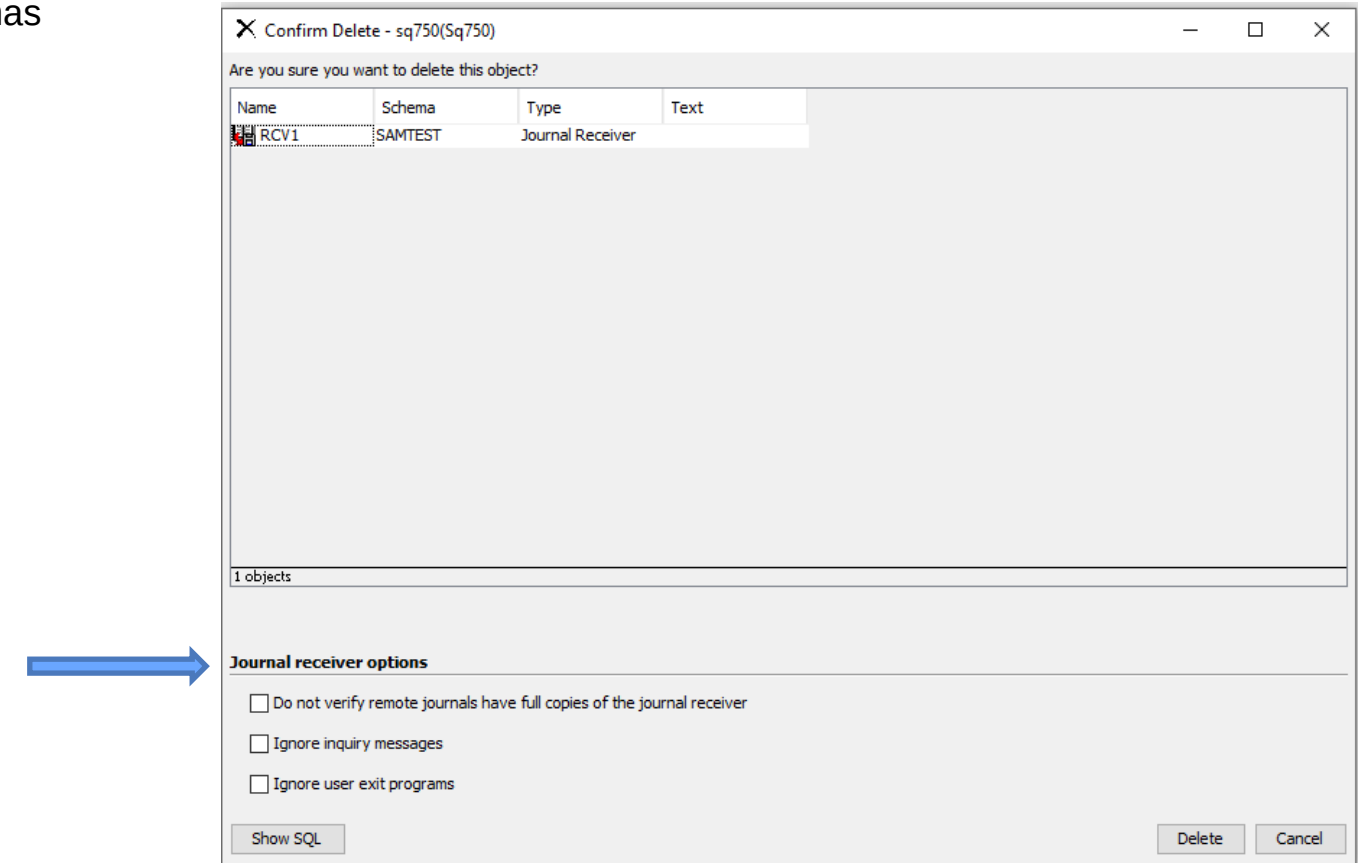


Run SQL scripts Protection



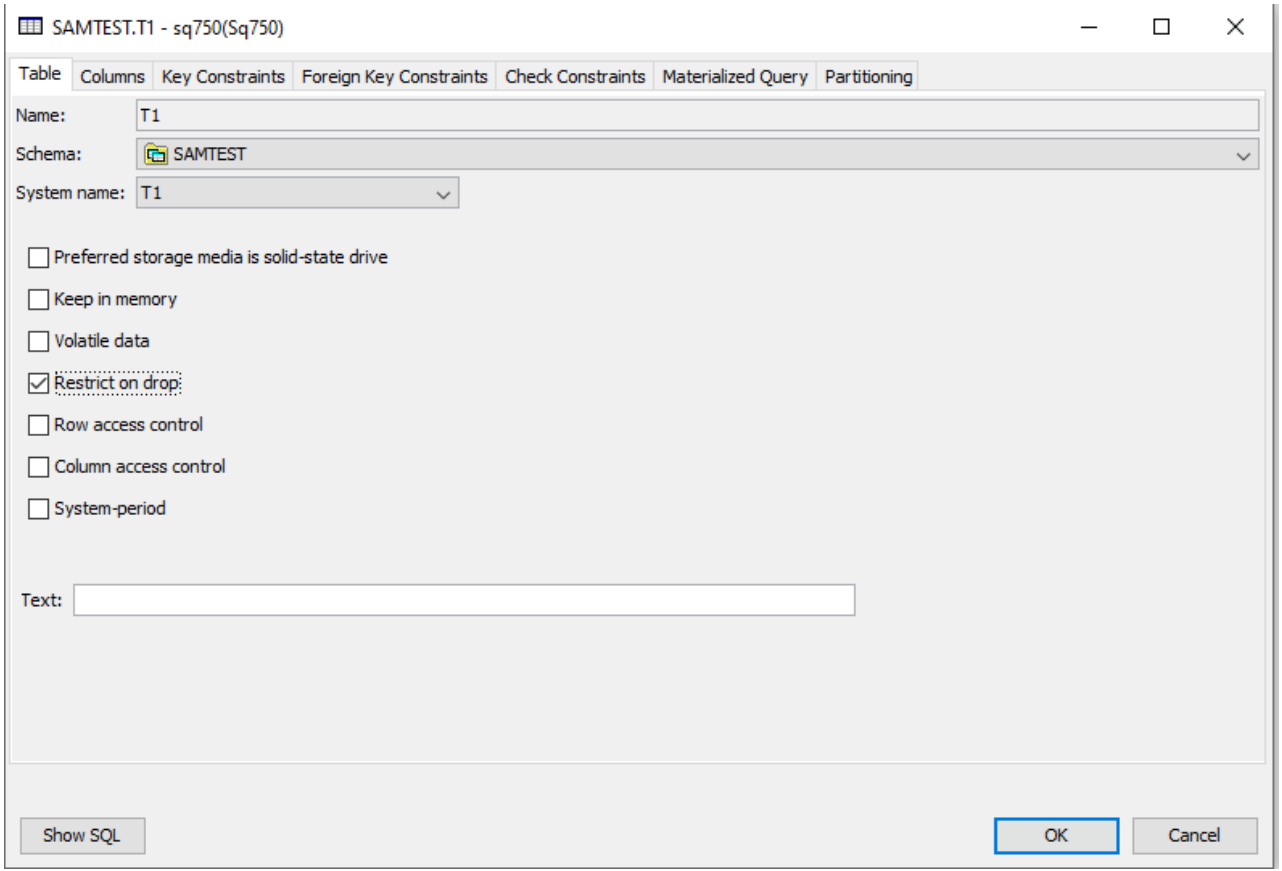
Schemas

- Provide the *IGNTGTRCV, *IGNINQMSG, *IGNEXITPGM options when deleting journal receivers from ACS Schemas



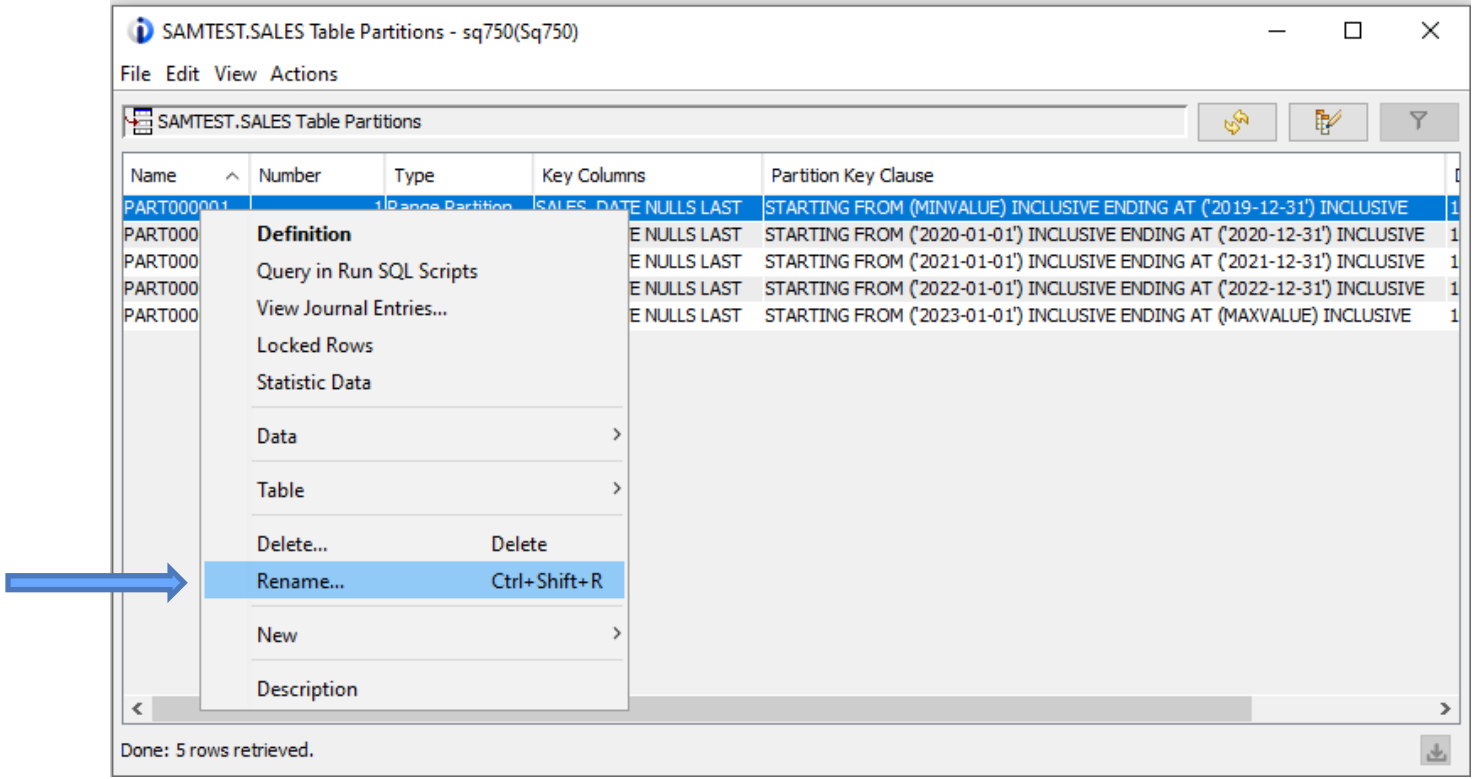
Schemas

- RESTRICT ON DROP



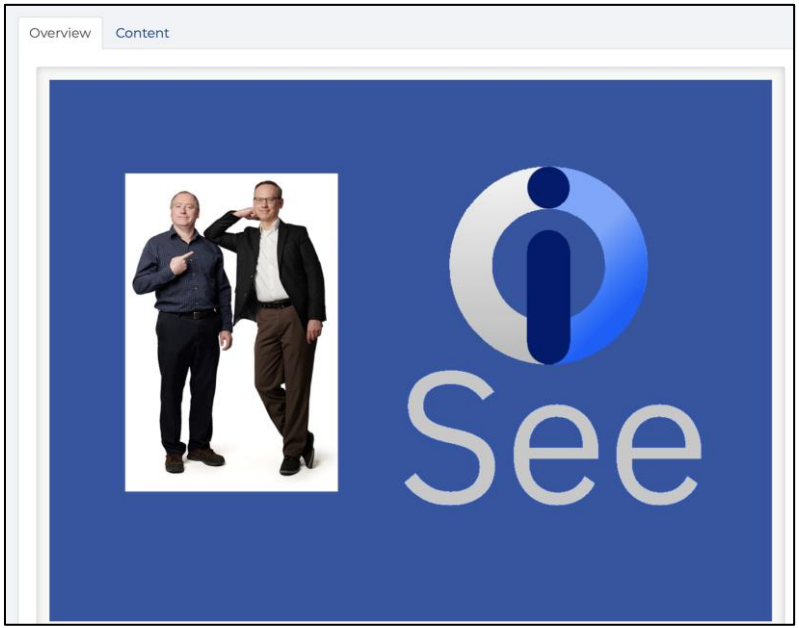
Schemas

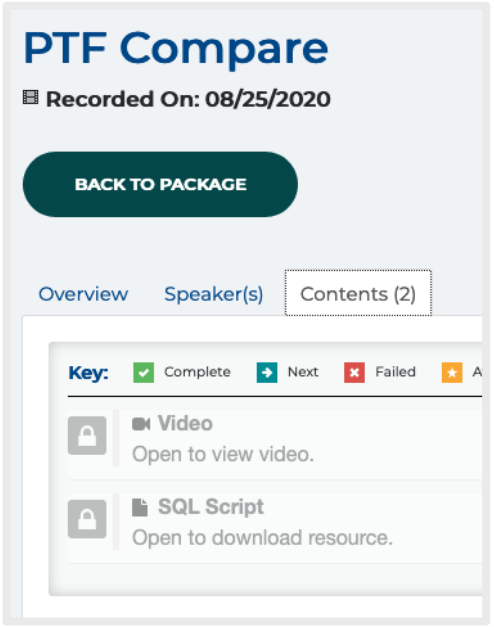
- Rename SQL Partitions



Find more information at …

iSee Video series

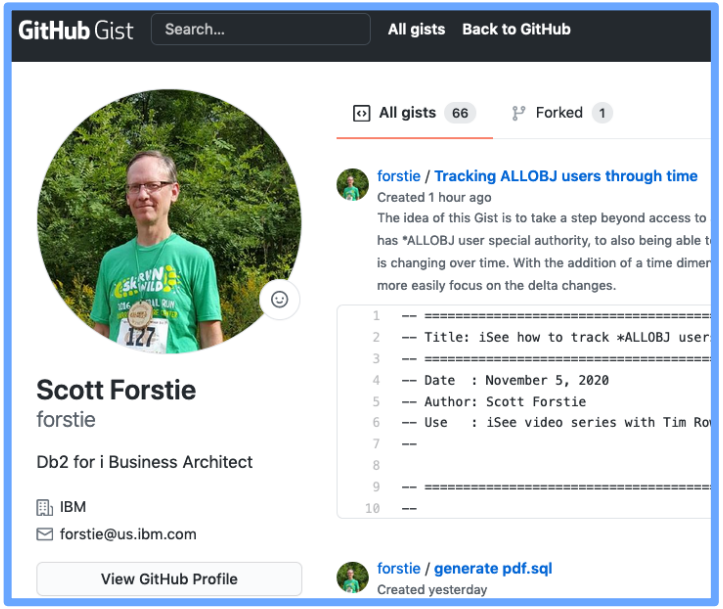




https://learn.common.org/products/ibm-isee#tab-product_tab_overview

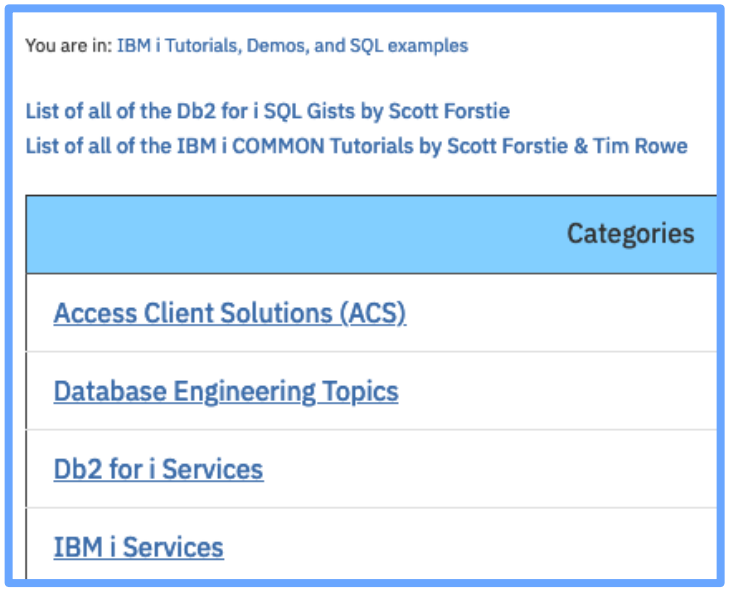
Find more information at …

SQL Examples published on Github



gist.github.com/forstie

Db2 for i SQL Tutor



CLASS IN SESSION

- Aggregation of all Gist Examples
- Aggregation of all iSee video blogs
- Different perspectives to easily find what you want

ibm.biz/Db2foriSQLTutor

IBM Technology Service Db2 for i Team

- SQL & Db2 Performance Assessment
  - Detailed analysis of SQL requests to identify performance bottlenecks
    - Index guidance that goes beyond the Db2 Index Advisor recommendations
    - SQL coding & configuration options also reviewed for performance inefficiencies
  - Includes a health check review of all Db2 objects
  - Client success stories
    - Recommended change to SQL delivered 1000x improvement for one report (your results may vary!)
    - Recommended index & configuration change dropped CPU utilization by 15%

- Database Engineering(DBE) Enablement – SQL Performance Tuning Workshop
  - Designed for both Admins and Developers:
    - Hands on education with ACS SQL Performance Center tooling
    - Best Practices for improving performance through indexing and more
    - SQL coding best practices to maximize performance
  - In person or remote customized education

For a complete list of Technology Services Db2 for i offerings: ibm.biz/Db2iTechnologyServices

For More Information:

Some Links You Need	Twitter	#Hashtags
IBM i Home Page: https://www.ibm.com/it-infrastructure/power/os/ibm-i IBM Strategy Whitepaper: https://www.ibm.com/it-infrastructure/us-en/resources/power/i-strategy-roadmap/ IBM Client Success: https://www.ibm.com/it-infrastructure/us-en/resources/power/ibm-i-customer-stories/ Support Life Cycle: https://www.ibm.com/support/lifecycle/ License Topics: https://www-01.ibm.com/support/docview.wss?uid=nas8N1022087 HelpSystems 2020 Marketplace Survey https://www.helpsystems.com/resources/guides/ibm-i-marketplace-survey-results	 @IBMSystems @COMMONug @IBMChampions @IBMSystemsISVs @IBMiMag @ITJungleNews @SAPonIBMi @SiDforIBMi	#PowerSystems #IBMi #IBMAIX #POWER9 #LinuxonPower #OpenPOWER #HANAonPower #ITinfrastructure #OpenSource #HybridCloud #BigData