Db2 Cheat Sheet for development

Created by: Andres Gomez Casanova
(@angoca)
Version: 2019-02-02
Get the most recent version at
https://github.com/angoca/db2-cheat-sheet/Db2CheatSheetForDev.pdf

Execution of a file in the console (db2clp)
- Semi-colon separated sentences:
```
db2 -t
```
- At sign separated sentences (when there is SQL PL code):
```
db2 -td@
```
Define a terminator character
```
--#SET TERMINATOR @
```
List all databases (aliases)
```
LIST DB DIRECTORY
```
Connect to a database (alias)
```
CONNECT TO mydb
```
Disconnect from a database
```
CONNECT RESET
TERMINATE
```
Get values from the environment (registry values)
- Current timestamp
```
VALUES CURRENT TIMESTAMP
```
- Connected user
```
VALUES CURRENT USER
```
- Current database
```
VALUES CURRENT SERVER
```
List all tables
```
LIST TABLES
LIST TABLES FOR SCHEMA myuser
LIST TABLES FOR ALL
```
Change current schema
```
SET CURRENT SCHEMA otherschema
```
Change the isolation level (RR, RS, CS, UR)
```
SET ISOLATION RR
```
List all tablespaces with their status
```
LIST TABLESPACES
```
Describe the structure of the table
```
DESCRIBE TABLE mytable
```
Describe the result of a query
```
DESCRIBE SELECT * FROM mytable
```
Get help for a Db2 command
```
? command
```
Get help for a SQL code (SQLXXXX) or SQLstate (YYYYY)
```
? SQLXXX
? YYYYY
```

DDL

Create a schema
```
CREATE SCHEMA myschema
```
Create a table in a specific tablespace
```
CREATE TABLE mytable1 (mycol1 SMALLINT NOT
  NULL, mycol2 VARCHAR(16)) IN ts1 INDEX
  IN ts2
CREATE TABLE myschema.othertable (mycol1
  SMALLINT)
```
Create a table like another one
```
CREATE TABLE mytable2 LIKE mytable1 IN ts1
  INDEX IN ts2
```
Comment on table and column
```
COMMENT ON TABLE mytable1 IS 'This is the
  comment of the table'
COMMENT ON COLUMN mytable1.mycol1 IS
  'Description of the field'
```
Declare a temporary table
```
DECLARE GLOBAL TEMPORARY TABLE mytemptab1
  (col1 SMALLINT, col2 TIMESTAMP)
```
Create a global temporary tablespace
```
CREATE GLOBAL TEMPORARY TABLE tmptable
  (col1 INTEGER)
```
Create an index
```
CREATE UNIQUE INDEX myidx2 ON mytable1
  (mycol2)
```
Add a column (requires Reorg table)
```
ALTER TABLE mytable1 ADD COLUMN mycol3
  timestamp
```
Change nullability
```
ALTER TABLE mytable1 ALTER COLUMN mycol3
  timestamp SET NOT NULL
```
Rename a column
```
ALTER TABLE mytable1 RENAME COLUMN mycol3
  TO mynewcol3
```
Delete the identify of a column
```
ALTER TABLE mytable1 TOODOOOOOOO
```
Create a primary key constraint
```
ALTER TABLE mytable1 ADD CONSTRAINT
  pkmytable PRIMARY KEY (mycol1)
```
Drop primary key
```
ALTER TABLE mytable1 DROP PRIMARY KEY
```
Add identity
```
ALTER TABLE mytable1 ALTER mycol1 SET
  GENERATED ALWAYS BY IDENTITY
```
Drop identity
```
ALTER TABLE mytable1 ALTER mycol1 DROP
  IDENTITY
```
Create a foreign key
```
ALTER TABLE mytable2 ADD CONSTRAINT
  fkmytable FOREIGN KEY (mycol1)
  REFERENCES mytable1 (mycol1)
```
Create a check constraint
```
ALTER TABLE mytable1 ADD CONSTRAINT chk
  CHECK (mycol2 in ('a', 'b', 'c'))
```
Enforce a constraint
```
ALTER TABLE mytable1 ALTER CHECK chk
  ENFORCED
```
Not enforce a constraint
```
ALTER TABLE mytable2 ALTER FOREIGN KEY
  fkmytable NOT ENFORCED
```
Change the granularity of the locks
```
ALTER TABLE mytable2 LOCKSIZE TABLE
```
Drop a table
```
DROP TABLE mytable
```
Rename a table
```
RENAME TABLE mytable2 AS myothertable
```
Truncate a table
```
TRUNCATE TABLE mytable1 IMMEDIATE
```
Create a sequence
```
CREATE SEQUENCE myseq AS INTEGER
```

Restart sequence

```
ALTER SEQUENCE myseq RESTART WITH 15
```

Crete a stored procedure

```
CREATE OR REPLACE PROCEDURE myproc (IN val
  SMALLINT, OUT ret VARCHAR(16)) SPECIFIC
  myproc1 BEGIN SET ret = (SELECT mycol2
  FROM mytable1 WHERE mycol1 = val); END @
```

Create a trigger

```
CREATE TRIGGER copy_value AFTER INSERT ON
  mytable1 REFERENCING NEW AS N FOR EACH
  ROW INSERT INTO mytable2 VALUES
  (N.mycol1, N.mycol2)
```

Create a view

```
CREATE VIEW VW1 AS SELECT mycol2 FROM
  mytable1
```

DCL

Grant on a table

```
GRANT SELECT, INSERT ON TABLE mytable TO
  GROUP recur
```

Grant execution on a stored procedure

```
GRANT EXECUTE ON PROCEDURE
  myproc(SMALLINT, VARCHAR(16)) TO USER
  jdoe
GRANT EXECUTE ON SPECIFIC PROCEDURE
  myproc1 TO USER jdoe
```

Revoke on a table

```
REVOKE UPDATE, DELETE ON TABLE mytable
  FROM GROUP recur
```

DML

Insert values on a table

```
INSERT INTO mytable1 VALUES (2, 'b')
INSERT INTO mytable1 VALUES (3, 'c'), (4,
  'd'), (5, 'e') --Atomic
```

Insert certain columns

```
INSERT INTO mytabl1 (mycol1) VALUES (6)
```

Insert values from a select

```
INSERT INTO myothertable SELECT mycol1,
  mycol2 FROM mytable1
```

Update fields

```
UPDATE mytable1 SET mycol1 = 5, mycol2 =
  'e' --all table
UPDATE mytable1 SET mycol2 = 'd' WHERE
  mycol1 = 7
```

Merge (upsert)

```
MERGE INTO mytable1 AS t USING (SELECT
  mycol1 FROM myothertable) s ON (t.mycol1
  = s.mycol1) WHEN MATCHED THEN UPDATE SET
  mycol2 = 'X' WHEN NOT MATCHED THEN
  INSERT VALUES (10, 'X')
```

Delete rows

```
DELETE FROM mytable1 --all table
DELETE FROM mytable1 WHERE mycol1 > 5
```

Export

```
EXPORT TO myfile OF DEL SELECT * FROM
  mytable1
IMPORT FROM myfile OF DEL INSERT INTO
  mytable1
```

Cursor

```
DECLARE mycursor CURSOR FOR SELECT * FROM
  mysrctable
```

Load

```
LOAD FROM myfile OF DEL INSERT INTO
  mytable1
LOAD FROM mycursor OF CURSOR INSERT INTO
  mytable1
```

Query the status of the load in a table

```
LOAD QUERY TABLE mytable1
```

Set integrity

```
SET INTEGRITY FOR mytable IMMEDIATE
  CHECKED
```

Ingest

```
INGEST FROM FILE my_file.txt FORMAT
  DELIMITED INSERT INTO my_table
```

Get the next value from a sequence

```
VALUES NEXT VALUE FOR myseq
INSERT INTO mytabl1 (mycol1) VALUES (NEXT
  VALUE FOR myseq)
```

TCL

Commit changes

```
COMMIT
```

Create a savepoint

```
SAVEPOINT sp1 ON ROLLBACK RETAIN CURSORS
```

Undo changes until savepoint

```
ROLLBACK TO SAVEPOINT sp1
```

Undo changes

```
ROLLBACK
```

Queries

Put a lock at table level

```
LOCK TABLE mytable1 IN EXCLUSIVE MODE
```

Execute a query without regard of commit rows

```
SELECT * FROM mytable WITH UR --RR,RS,CS
```

Execute a query with only 5 rows

```
SELECT * FROM mytable FETCH FIRST 5 ROWS
  ONLY
```

Perform a query to a dummy table (dual)

```
SELECT 'Any string' FROM SYSIBM.SYSDUMMY1
```

Perform a query calling a function

```
SELECT HEX(mycol2) FROM mytable1
```

Call a function

```
VALUES HEX('AnyText')
```

Perform a cast

```
VALUES CAST('123' AS INTEGER)
```

Concatenate

```
VALUES 'AnyText' || 5
VALUES 'AnyText' concat 5
```

Escape a single quote in a text field

```
VALUES 'Sinead o''Connor'
```

Query the database catalog

```
SELECT * FROM SYSCAT.TABLES
SELECT * FROM SYSCAT.TABAUTH
SELECT * FROM SYSCAT.ROUTINES
```

SQL PL

Create a compound statement – Anonymous block

```
BEGIN DECLARE val SMALLINT; SET val = 1;
  WHILE (val <= 5) DO INSERT INTO mytable
  VALUES (val, val); SET val = val + 1;
  END WHILE; END @
```

Perform a reorg via ADMIN_CMD (Sometimes required after “alter table”)

```
CALL SYSPROC.ADMIN_CMD('REORG TABLE
  mytable')
```

Call a stored procedure with an IN and an OUTPUT parameter

```
CALL myproc(5, ?)
```