SQL Cheat Sheet: Views, Stored Procedures and Transactions

Views

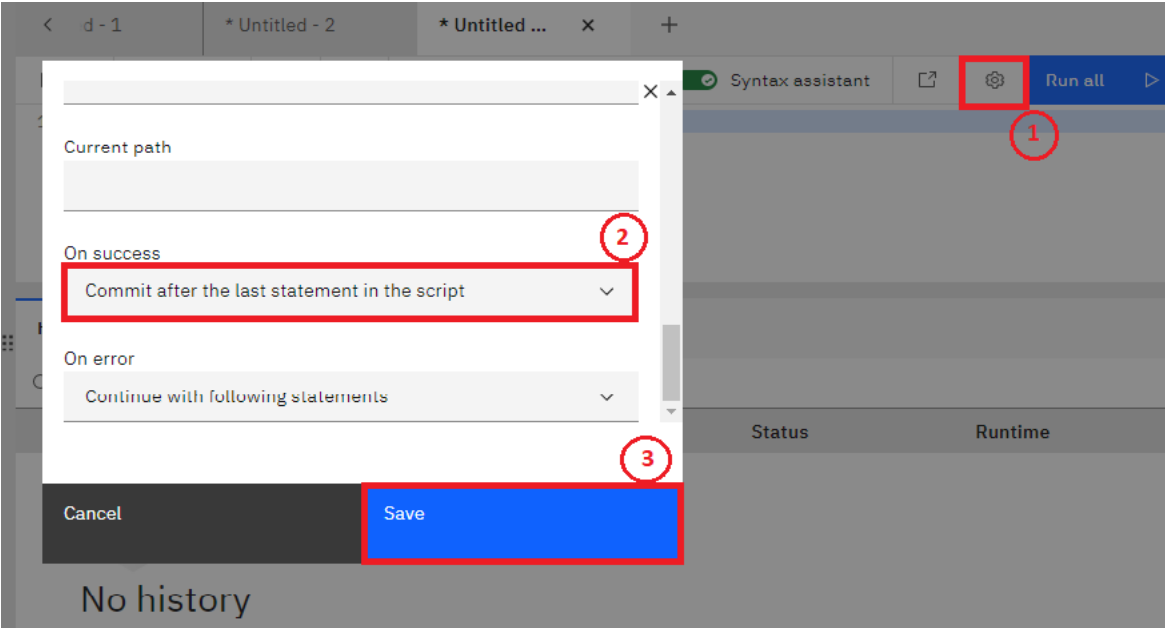| Topic	Syntax	Description	Example
Create View	CREATE VIEW view_name AS SELECT column1, column2, ... FROM table_name WHERE condition;	A CREATE VIEW is an alternative way of representing data that exists in one or more tables.	CREATE VIEW EMPSALARY AS SELECT EMP_ID, F_NAME, L_NAME, B_DATE, SEX, SALARY FROM EMPLOYEES;
Update a View	CREATE OR REPLACE VIEW view_name AS SELECT column1, column2, ... FROM table_name WHERE condition;	The CREATE OR REPLACE VIEW command updates a view.	CREATE OR REPLACE VIEW EMPSALARY AS SELECT EMP_ID, F_NAME, L_NAME, B_DATE, SEX, JOB_TITLE, MIN_SALARY, MAX_SALARY FROM EMPLOYEES, JOBS WHERE EMPLOYEES.JOB_ID = JOBS.JOB_IDENT;
Drop a View	DROP VIEW view_name;	Use the DROP VIEW statement to remove a view from the database.	DROP VIEW EMPSALARY;

Stored Procedures in IBM Db2 using SQL

Topic	Syntax	Description	Example
Stored Procedures	--#SET TERMINATOR @ CREATE PROCEDURE PROCEDURE_NAME LANGUAGE BEGIN END @	A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. The default terminator for a stored procedure is semicolon(;). To set a different terminator we use SET TERMINATOR clause followed by the terminator such as '@'.	--#SET TERMINATOR @ CREATE PROCEDURE RETRIEVE_ALL LANGUAGE SQL READS SQL DATA DYNAMIC RESULT SETS 1 BEGIN DECLARE C1 CURSOR WITH RETURN FOR SELECT * FROM PETSALE; OPEN C1; END @

Stored Procedures in MySQL using phpMyAdmin

Topic	Syntax	Description	Example
Stored Procedures	DELIMITER // CREATE PROCEDURE PROCEDURE_NAME BEGIN END // DELIMITER ;	A stored procedure is a prepared SQL code that you can save, so the code can be reused over and over again. The default terminator for a stored procedure is semicolon (;). To set a different terminator we use DELIMITER clause followed by the terminator such as $$ or //.	DELIMITER // CREATE PROCEDURE RETRIEVE_ALL() BEGIN SELECT * FROM PETSALE; END // DELIMITER ;

Transactions with Db2

Topic	Syntax	Description	Example
Commit command	COMMIT;	A COMMIT command is used to persist the changes in the database. The default terminator for a COMMIT command is semicolon (;).	CREATE TABLE employee(ID INT, Name VARCHAR(20), City VARCHAR(20), Salary INT, Age INT); INSERT INTO employee(ID, Name, City, Salary, Age) VALUES(1, 'Priyanka pal', 'Nasik', 36000, 21), (2, 'Riya chowdary', 'Ba 82000, 29); SELECT *FROM employee; COMMIT;
Rollback command	ROLLBACK;	A ROLLBACK command is used to rollback the transactions which are not saved in the database. The default terminator for a ROLLBACK command is semicolon (;).	As auto-commit is enabled by default, all transactions will be committed. We need to disable this option to see how rollback works. For db2, we have to disable auto-commit manually. Click the gear icon located on the right side of the SQL Assistant window. Next, select the "On Success" drop-down and choose "commit after the last statement in the script" Remember t save your changes!

```
INSERT INTO employee VALUES (3, 'Swetha Tiwari', 'Kanpur', 38000, 38);

SELECT *FROM employee;
ROLLBACK;
SELECT *FROM employee;
```

Transactions with MySQL

Commit command	`COMMIT;`	A `COMMIT` command is used to persist the changes in the database.
		
		The default terminator for a COMMIT command is semicolon (;).
			```CREATE TABLE employee(ID INT, Name VARCHAR(20), City VARCHAR(20), Salary INT, Age INT);```
			```START TRANSACTION;```
			```INSERT INTO employee(ID, Name, City, Salary, Age) VALUES(1, 'Priyanka pal', 'Nasik', 36000, 21), (2, 'Riya chowdary', 'Bangalor', 82000, 29);```
			```SELECT *FROM employee;```
			```COMMIT;```
			
			    As auto-commit is enabled by default, all transactions will be committed. We need to disable this option to see how rollback works. For MySQL use the command "SET autocommit = 0;"

Rollback command	`ROLLBACK;`	A `ROLLBACK` command is used to rollback the transactions which are not saved in the database.
		
		The default terminator for a ROLLBACK command is semicolon (;).
			```INSERT INTO employee VALUES (3, 'Swetha Tiwari', 'Kanpur', 38000, 38);```
			
			```SELECT *FROM employee;```
			```ROLLBACK;```
			```SELECT *FROM employee;```

Db2 Transactions using Stored Procedure

Commit command	```–#SET TERMINATOR @```
	
	```CREATE PROCEDURE PROCEDURE_NAME```
	
	```BEGIN```
	
	```COMMIT;```
	
	```END```
	```@```
		A `COMMIT` command is used to persist the changes in the database.
		
		The default terminator for a COMMIT command is semicolon (;).
			```--#SET TERMINATOR @ CREATE PROCEDURE TRANSACTION_ROSE LANGUAGE SQL MODIFIES SQL DATA```
			
			```BEGIN```
			
			```DECLARE SQLCODE INTEGER DEFAULT 0;```
			```DECLARE retcode INTEGER DEFAULT 0;```
			```DECLARE CONTINUE HANDLER FOR SQLEXCEPTION SET retcode = SQLCODE;```
			
			```UPDATE BankAccounts```
			```SET Balance = Balance-200```
			```WHERE AccountName = 'Rose';```
			
			```UPDATE BankAccounts```
			```SET Balance = Balance-300```
			```WHERE AccountName = 'Rose';```
			
			```IF retcode < 0 THEN```
			```ROLLBACK WORK;```
			
			```ELSE```
			```COMMIT WORK;```
			
			```END IF;```

	--#SET TERMINATOR @ CREATE PROCEDURE PROCEDURE_NAME BEGIN	
Rollback command	ROLLBACK; COMMIT; END @	A `ROLLBACK` command is used to rollback the transactions which are not saved in the database. The default terminator for a ROLLBACK command is semicolon (;).

```
END
@
--#SET TERMINATOR @ CREATE PROCEDURE
TRANSACTION_ROSE LANGUAGE SQL MODIFIES SQL
DATA

BEGIN

DECLARE SQLCODE INTEGER DEFAULT 0;
DECLARE retcode INTEGER DEFAULT 0;
DECLARE CONTINUE HANDLER FOR SQLEXCEPTION
SET retcode = SQLCODE;

UPDATE BankAccounts
SET Balance = Balance-200
WHERE AccountName = 'Rose';

UPDATE BankAccounts
SET Balance = Balance-300
WHERE AccountName = 'Rose';

IF retcode < 0 THEN
ROLLBACK WORK;

ELSE
COMMIT WORK;

END IF;

END
@
```

MySQL Transactions using Stored Procedure

Commit command	DELIMITER // CREATE PROCEDURE PROCEDURE_NAME BEGIN COMMIT; END // DELIMITER ;	A `COMMIT` command is used to persist the changes in the database. The default terminator for a COMMIT command is semicolon (;).

```
DELIMITER //

CREATE PROCEDURE TRANSACTION_ROSE()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION
BEGIN
ROLLBACK;
RESIGNAL;
END;

START TRANSACTION;
UPDATE BankAccounts
SET Balance = Balance-200
WHERE AccountName = 'Rose';

UPDATE BankAccounts
SET Balance = Balance-300
WHERE AccountName = 'Rose';

COMMIT;

END //

DELIMITER ;
```

Rollback command	DELIMITER // CREATE PROCEDURE PROCEDURE_NAME BEGIN ROLLBACK; COMMIT; END // DELIMITER ;	A `ROLLBACK` command is used to rollback the transactions which are not saved in the database. The default terminator for a ROLLBACK command is semicolon (;).

```
DELIMITER //

CREATE PROCEDURE TRANSACTION_ROSE()

BEGIN

DECLARE EXIT HANDLER FOR SQLEXCEPTION
BEGIN
ROLLBACK;
RESIGNAL;
END;

START TRANSACTION;
UPDATE BankAccounts
SET Balance = Balance-200
WHERE AccountName = 'Rose';

UPDATE BankAccounts
SET Balance = Balance-300
WHERE AccountName = 'Rose';

COMMIT;

END //

DELIMITER ;
```

Author(s)

D.M Naidu

Skills Network

Changelog

Date	Version	Changed by	Change Description
2022-10-04	1.0	D.M.Naidu	Initial Version