SQL Cheat Sheet: Views, Stored Procedures, Transactions and JOIN statements

IBM Developer
SKILLS NETWORK

Topic	Syntax	Description	Example
Create View	`CREATE VIEW view_name AS SELECT column1, column2, ... FROM table_name WHERE condition;`	A `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 on IBM Db2 using SQL

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

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 ;`

Joins

Cross Join	`SELECT column_name(s) FROM table1 CROSS JOIN table2;`	The `CROSS JOIN` is used to generate a paired combination of each row of the first table with each row of the second table.	`SELECT DEPT_ID_DEP, LOCT_ID FROM DEPARTMENTS CROSS JOIN LOCATIONS;`
Inner Join	`SELECT column_name(s) FROM table1 INNER JOIN table2 ON table1.column_name = table2.column_name; WHERE condition;`	You can use an `inner join` in a SELECT statement to retrieve only the rows that satisfy the join conditions on every specified table.	`select E.F_NAME,E.L_NAME, JH.START_DATE from EMPLOYEES as E INNER JOIN JOB_HISTORY as JH on E.EMP_ID=JH.EMPL_ID where E.DEP_ID ='5';`
Left Outer Join	`SELECT column_name(s) FROM table1 LEFT OUTER JOIN table2 ON table1.column_name = table2.column_name WHERE condition;`	The `LEFT OUTER JOIN` will return all records from the left side table and the matching records from the right table.	`select E.EMP_ID,E.L_NAME,E.DEP_ID,D.DEP_NAME from EMPLOYEES AS E LEFT OUTER JOIN DEPARTMENTS AS D ON E.DEP_ID=D.DEPT_ID_DEP;`
Right Outer Join	`SELECT column_name(s) FROM table1 RIGHT OUTER JOIN table2 ON table1.column_name = table2.column_name WHERE condition;`	The `RIGHT OUTER JOIN` returns all records from the right table, and the matching records from the left table.	`select E.EMP_ID,E.L_NAME,E.DEP_ID,D.DEP_NAME from EMPLOYEES AS E RIGHT OUTER JOIN DEPARTMENTS AS D ON E.DEP_ID=D.DEPT_ID_DEP;`
Full Outer Join	`SELECT column_name(s) FROM table1 FULL OUTER JOIN table2 ON table1.column_name = table2.column_name WHERE condition;`	The `FULL OUTER JOIN` clause results in the inclusion of rows from two tables. If a value is missing when rows are joined, that value is null in the result table.	`select E.F_NAME,E.L_NAME,D.DEP_NAME from EMPLOYEES AS E FULL OUTER JOIN DEPARTMENTS AS D ON E.DEP_ID=D.DEPT_ID_DEP;`
Self Join	`SELECT column_name(s) FROM table1 T1, table1 T2 WHERE condition;`	A `self join` is regular join but it can be used to joined with itself.	`SELECT B.* FROM EMPLOYEES A JOIN EMPLOYEES B ON A.MANAGER_ID = B.MANAGER_ID WHERE A.EMP_ID = 'E1001';`

Joins in MySQL using phpMyAdmin

Full Outer Join	`SELECT column_name(s) FROM table1 LEFT OUTER JOIN table2 ON table1.column_name = table2.column_name WHERE condition`
	`UNION`
	`SELECT column_name(s) FROM table1 RIGHT OUTER JOIN table2 ON table1.column_name = table2.column_name WHERE condition`
		The `UNION operator` is used to combine the result-set of two or more SELECT statements.	`select E.F_NAME,E.L_NAME,D.DEP_NAME from EMPLOYEES AS E LEFT OUTER JOIN DEPARTMENTS AS D ON E.DEP_ID=D.DEPT_ID_DEP`
			`UNION`
			`select E.F_NAME,E.L_NAME,D.DEP_NAME from EMPLOYEES AS E RIGHT OUTER JOIN DEPARTMENTS AS D ON E.DEP_ID=D.DEPT_ID_DEP`

Author(s)

D.M Naidu

Changelog

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