# MySQL Differences with Oracle

# Documentation

Reference Manual:

<https://dev.mysql.com/doc/refman/8.0/en/>

# Tools and Programs

Overview of all programs : <https://dev.mysql.com/doc/refman/8.0/en/programs-overview.html>'

## Command LineTools mysql (SQL\*Plus)

mysql is a command line client similar to sqlplus

## MySQLWorkbench ( SQL\*Developer)

MySQLWorkbench desktop GUI ( and looks like Windows-only) client similar to SQL\*Developer

## Server Daemon

**Mysqld – is database engine executable i.e. MySQL Server**

# Maintenance

## Start/Stop On Windows

Server executable is **mysqld.exe**

The best way to start stop seems to bounce MySQL service:

|  |
| --- |
| net start MySQL80  net stop MySQL80 |

**Note**: Obviously from admin account

Documented way to start execrable and pass .ini file for some **reason did not work and does not produce any output**!

|  |
| --- |
| "C:\Program Files\MySQL\MySQL Server 8.0\bin\mysqld.exe" --defaults-file="C:\ProgramData\MySQL\MySQL Server 8.0\my.ini" MySQL80 |

## Start/Stop On Unix ( NOT Verified!)

Service/start stop on Linux can be done like this:

|  |
| --- |
| sudo service mysql start  sudo service mysql stop |

## Parameters file my.ini (i.e. init.ora)

Server starts with ini file passed in –defaults-file parameter

|  |
| --- |
| mysqld --defaults-file="C:\Program Files\MySQL\MySQL Server X.Y\my.ini" |

## Datafiles Location and Structure

Datafile located in a folder specified by datadir parameter in my.ini:

|  |
| --- |
| # Path to the database root  datadir="C:/ProgramData/MySQL/MySQL Server 8.0/Data" |

It looks like datafiles are structured as following:

* Schema – folder
* Table – file in the folder with ibd extension
* I.e. schema1/table1 data are located in ${datadir}/ schema1/table1.bid

## Log Files (i.e. alert.log)

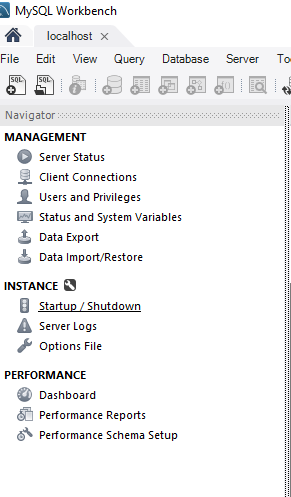
Logs are located in ${datadir}. Names and content are defined by parameters in my.ini

|  |
| --- |
| # General and Slow logging.  log-output=FILE  general-log=0  general\_log\_file="ALEX-XPS15-9560.log"  slow-query-log=1  slow\_query\_log\_file="ALEX-XPS15-9560-slow.log"  long\_query\_time=10  # Error Logging.  log-error="ALEX-XPS15-9560.err" |

**log-error is the closest thing to Oracle alert.log**  as it has records about stop and start

## GUI Maintenance Tools

MySQLWorkbench has GUI maintenance tools under Administration tab



# Command Line Tool mysql (i.e. SQL\*Plus)

## Command Line Options

<https://dev.mysql.com/doc/refman/8.0/en/mysql-command-options.html>

## Help

To see mysql help:

|  |
| --- |
| mysql> help;  For information about MySQL products and services, visit:  http://www.mysql.com/  For developer information, including the MySQL Reference Manual, visit:  http://dev.mysql.com/  To buy MySQL Enterprise support, training, or other products, visit:  https://shop.mysql.com/  List of all MySQL commands:  Note that all text commands must be first on line and end with ';'  ? (\?) Synonym for `help'.  clear (\c) Clear the current input statement.  connect (\r) Reconnect to the server. Optional arguments are db and host.  delimiter (\d) Set statement delimiter.  ego (\G) Send command to mysql server, display result vertically.  exit (\q) Exit mysql. Same as quit.  go (\g) Send command to mysql server.  help (\h) Display this help.  notee (\t) Don't write into outfile.  print (\p) Print current command.  prompt (\R) Change your mysql prompt.  quit (\q) Quit mysql.  rehash (\#) Rebuild completion hash.  source (\.) Execute an SQL script file. Takes a file name as an argument.  status (\s) Get status information from the server.  system (\!) Execute a system shell command.  tee (\T) Set outfile [to\_outfile]. Append everything into given outfile.  use (\u) Use another database. Takes database name as argument.  charset (\C) Switch to another charset. Might be needed for processing binlog with multi-byte charsets.  warnings (\W) Show warnings after every statement.  nowarning (\w) Don't show warnings after every statement.  resetconnection(\x) Clean session context. |

## Verbose ( i.e. Echo on)

-v in command line

|  |
| --- |
| C:\Program Files\MySQL\MySQL Server 8.0\bin>mysql -v -u root -ppassword join\_us < C:/git/mysql/commands/bind\_var.sql  mysql: [Warning] Using a password on the command line interface can be insecure.  --------------  set @num\_users := 0  --------------  --------------  select @num\_users  --------------  @num\_users  0  --------------  select count(\*) into @num\_users from users  --------------  --------------  select @num\_users  --------------  @num\_users  1517 |

## System (i.e. host)

Executes OS command

|  |
| --- |
| For server side help, type 'help contents'  mysql> system dir;  Volume in drive C is OS  Volume Serial Number is 0867-9728  Directory of C:\Program Files\MySQL\MySQL Server 8.0\bin |

## Source (i.e. @)

### From inside mysql

**Note "/" for Windows!**

source C:/git/mysql/commands/bind\_var.sql;

### From Command Line

Use redirect:

|  |
| --- |
| pushd "C:\Program Files\MySQL\MySQL Server 8.0\bin\"  mysql –v u root -ppassword join\_us < C:/git/mysql/commands/bind\_var.sql |

## Tee ( i.e. spool)

**Note "/" in path for Windows!**

|  |
| --- |
| tee C:/git/mysql/commands/spool.log;  source C:/git/mysql/commands/bind\_var.sql;  notee; |

Generates spool.log:

|  |
| --- |
| mysql> tee C:/git/mysql/commands/spool.log;source C:/git/mysql/commands/bind\_var.sql;notee;  Logging to file 'C:/git/mysql/commands/spool.log'  Query OK, 0 rows affected (0.00 sec)  +------------+  | @num\_users |  +------------+  | 0 |  +------------+  1 row in set (0.00 sec)  Query OK, 1 row affected (0.00 sec)  +------------+  | @num\_users |  +------------+  | 1517 |  +------------+  1 row in set (0.00 sec)  Outfile disabled. |

## Delimiters

Delimiter should be set to something other than semicolon to allow "compound statements like CREATE PROCEDURE or CREATE TRIGGER"

|  |
| --- |
| delimiter //  drop function hello//  create function hello ( p\_name varchar(100))  returns varchar(100) deterministic  begin  return concat ('Hello ,',p\_name, '!' );  end;//  delimiter ;  select t1.\* , hello(t1.f1) from t1 ; |

## Scripts Error Handling

# STRICT MODE and Warnings

# SQL

## Key Minuses

* No FULL OUTER JOIN
* No simple data arithmetic
* No || string operator ( concat is used instead)
* Collation rules ( string comparison ) are very complicated
* Idiotic issues with space before ()

## Key Pluses

* Easy use of Boolean

select A=B; -- returns 0 or 1

* Multiple INSERT

## Comments

* -- Line comments must have space after two dashes
* /\* \*/ like in Oracle
* # also allowed

## Bind Variables

* Bind variable are prefixed by @
* SELECT INTO … @bind\_var does not have to be inside begin … end

|  |
| --- |
| set @num\_users := 0;  select @num\_users;  select count(\*) into @num\_users from users;  select @num\_users; |

## Issue with Parentheses

It can drive me crazy there should be no space between function name and () BUT ONLY for dictionary functions ….

|  |
| --- |
| mysql> select now();  +---------------------+  | now() |  +---------------------+  | 2020-09-08 16:12:51 |  +---------------------+  1 row in set (0.00 sec)  mysql> select now ();  ERROR 1630 (42000): FUNCTION join\_us.now does not exist. Check the 'Function Name Parsing and Resolution' section in the Reference Manual  mysql> select hello ('a');  +-------------+  | hello ('a') |  +-------------+  | Hello ,a! |  +-------------+  1 row in set (0.00 sec)  mysql> select hello('a');  +------------+  | hello('a') |  +------------+  | Hello ,a! |  +------------+  1 row in set (0.00 sec) |

## Quotation

### Data

Both single and double quotes are accepted

|  |
| --- |
| -- quotes in data  drop table if exists t3;    create table t3( name varchar(100));  insert into t3 (name)values ('single quites') ;  insert into t3 (name)values ("double quites") ;  insert into t3 (name)  values  ("single 'quote' inside double"),  ('double "quote" inside single'),  ('single \'quote\' inside single'),  ("double \"quote\" inside double")  ;  select \* from t3; |

### Metadata

Backquote ` us used for metadata quotes

|  |
| --- |
| -- metadata quotes  drop table if exists `table with space`;  create table if not exists `table with space`  ( `col with space` varchar(100));    insert into `table with space` ( `col with space`)  select name from t3;  select distinct `col with space` from `table with space`; |

# Transactional Control

<https://dev.mysql.com/doc/refman/8.0/en/commit.html>

## Autocommit

Autocommit is enabled by default

To disable autocommit mode explicitly, use the following statement:

SET autocommit=0;

## Start Transaction

To disable autocommit mode implicitly for a single series of statements, use the START TRANSACTION statement

|  |
| --- |
| SET autocommit=1;  insert into t1 values ('d', 'd');  /\* visible from the other session right away \*/  START TRANSACTION;  insert into t1 values ('e', 'e');  /\* the other session waits \*/  rollback; |

# IF EXISTS and IF NOT EXISTS Clauses

CREATE /DROP table, database . trigger and etc have EXISTS / NOT EXISTS clauses that give a warning ( NOT ERROR ) if object on question does or does not exists:

|  |
| --- |
| mysql> create table if not exists t1( f1 varchar(255) primary key, f2 varchar(255) unique key );  Query OK, 0 rows affected, 1 warning (0.00 sec)  mysql> show warnings  -> ;  +-------+------+---------------------------+  | Level | Code | Message |  +-------+------+---------------------------+  | Note | 1050 | Table 't1' already exists |  +-------+------+---------------------------+  1 row in set (0.00 sec)  mysql> drop table if exists t123456;  Query OK, 0 rows affected, 1 warning (0.01 sec)  mysql> show warnings;  +-------+------+---------------------------------+  | Level | Code | Message |  +-------+------+---------------------------------+  | Note | 1051 | Unknown table 'join\_us.t123456' |  +-------+------+---------------------------------+  1 row in set (0.00 sec) |

# LIMIT ( i.e ROWNUM )

# INFORMATION\_SCHEMA.(i.e. Data Dictionary)

Looks like data dictionary views equivalent to are ALL\_TABLES, ALL\_VIEWS and etc. listed in INFORMATION\_SCHEMA.tables

|  |
| --- |
| SELECT \* FROM INFORMATION\_SCHEMA.tables where table\_schema = 'INFORMATION\_SCHEMA'; |

For example this show all triggers in the current database

|  |
| --- |
| SELECT \* FROM INFORMATION\_SCHEMA.triggers t where t.EVENT\_OBJECT\_SCHEMA = database() ; |

# Databases (i.e. Schemas)

## Show Databases

## Create

## Use Database

## Show Current Database

## Drop

|  |
| --- |
| show databases;  create database if not exists test;  use test;  select database();  create table t1  as  select \* from join\_us.t1;  select \* from test.t1;    drop database if exists test; |

# NULLS

## Empty String IS NOT NULL

## IFNULL (i.e. NVL)

|  |
| --- |
| drop table if exists n1;  create table n1  ( f1 varchar(100),  cmt varchar(100)  );  insert into n1 values ( '', 'emptry string'), ( null, 'NULL');  select \* from n1;  select \* from n1 where f1 is null;    select \* from n1 where f1 = "";  select \* from n1 where ifnull(f1,"") = ""; |

# DUAL (i.e. One Row SQL)

Select anything without from i.e.

|  |
| --- |
| mysql> Select database(), 13+45;  +------------+-------+  | database() | 13+45 |  +------------+-------+  | join\_us | 58 |  +------------+-------+  1 row in set (0.00 sec) |

# INSERT

## INSERT Multiple Inserts

|  |
| --- |
| INSERT INTO cats(name, breed, age)  VALUES ('Ringo', 'Tabby', 4),  ('Cindy', 'Maine Coon', 10),  ('Dumbledore', 'Maine Coon', 11),  ('Egg', 'Persian', 4),  ('Misty', 'Tabby', 13),  ('George Michael', 'Ragdoll', 9),  ('Jackson', 'Sphynx', 7); |

## INSERT SET

## INSERT IGNORE

<https://dev.mysql.com/doc/refman/8.0/en/sql-mode.html#ignore-effect-on-execution>

With IGNORE, the row containing the duplicate key still is not inserted, but a warning occurs instead of an error

[INSERT](https://dev.mysql.com/doc/refman/8.0/en/insert.html): With IGNORE, rows that duplicate an existing row on a unique key value are discarded. Rows set to values that would cause data conversion errors are set to the closest valid values instead.

|  |
| --- |
| Mysql> select \* from cats;  | cat\_id | name | breed | age |  | 1 | Ringo | Tabby | 4 |  | 2 | Cindy | Maine Coon | 10 |  | 3 | Dumbledore | Maine Coon | 11 |  | 4 | Egg | Persian | 4 |  | 5 | Misty | Tabby | 13 |  | 6 | George Michael | Ragdoll | 9 |  | 7 | Jackson | Sphynx | 7 |  | 8 | Pussy | New | NULL |  8 rows in set (0.00 sec)  mysql> INSERT INTO cats(cat\_id, name, breed, age) VALUES (1, 'Ringo', 'Tabby', 4);  ERROR 1062 (23000): Duplicate entry '1' for key 'cats.PRIMARY'  mysql>  mysql> INSERT IGNORE INTO cats(cat\_id, name, breed, age) VALUES (1, 'Ringo', 'Tabby', 4);show warnings;  Query OK, 0 rows affected, 1 warning (0.00 sec)  | Level | Code | Message |  | Warning | 1062 | Duplicate entry '1' for key 'cats.PRIMARY' |  1 row in set (0.00 sec)  mysql> select \* from cats;  | cat\_id | name | breed | age |  | 1 | Ringo | Tabby | 4 |  | 2 | Cindy | Maine Coon | 10 |  | 3 | Dumbledore | Maine Coon | 11 |  | 4 | Egg | Persian | 4 |  | 5 | Misty | Tabby | 13 |  | 6 | George Michael | Ragdoll | 9 |  | 7 | Jackson | Sphynx | 7 |  | 8 | Pussy | New | NULL |  8 rows in set (0.00 sec) |

# Aggregate Functions

# AUTO INCREMENT (i.e. Sequences )

# Primary/Unique Keys on Strings

# Boolean

Boolean expressions

# Dates

## Datatypes

## Functions

## Sysdate() VS. NOW()

## Casting

## Date Math

# Strings

## Case Sensitivity

The default character set and collation are utf8mb4 and utf8mb4\_0900\_ai\_ci, so nonbinary string comparisons are case-insensitive by default. This means that if you search with ***col\_name*** LIKE 'a%', you get all column values that start with A or a. To make this search case-sensitive, make sure that one of the operands has a case-sensitive or binary collation. For example, if you are comparing a column and a string that both have the utf8mb4 character set, you can use the COLLATE operator to cause either operand to have the utf8mb4\_0900\_as\_cs or utf8mb4\_bin collation:

<https://dev.mysql.com/doc/refman/8.0/en/case-sensitivity.html>

Case Insensitive Comparison

|  |
| --- |
| create table if not exists t1  ( f1 varchar(255) primary key,  f2 varchar(255) unique key  );    insert into t1 values ("a", "a");  -- match  select \* from t1 where f1 like 'A%';  -- CS no match  select \* from t1 where f1 COLLATE utf8mb4\_0900\_as\_cs like 'A%';  -- CI match  select \* from t1 where f1 COLLATE utf8mb4\_0900\_as\_ci like 'A%';  -- BIN no match  select \* from t1 where f1 COLLATE utf8mb4\_bin like 'A%'; |

## Strings Datatypes

## Strings Functions

# Compound Statements (i.e. PL/SQL)

Compound statements provide a procedural language somewhat similar to PL/SQL

## Key Limitations:

* No packages and package variables
* No DBMS\_OUTPUT equivalent
* No anonymous PL/SQL Blocks
* No amomums transactions
* No passing arrays to stored procedures;
* Compilation errors like undefined variables are thrown on runtime;
* No CREATE OR REPLACE
* Signal Message cannot be an expression (must be either variable or constant)

## SQLSTATE (i.e. SQLCODE)

SQLSTATE [VALUE] ***sqlstate\_value***: A 5-character string literal indicating an SQLSTATE value.

Similar to use to Oracle SQLCODE

## Signal and Resignal (i.e. RAISE and RAISE\_APPLICATION\_ERROR)

### Custom errors i.e. RAISE\_APPLICATION\_ERROR

Custom errors must use sqlstate '45000'

|  |
| --- |
| declare err\_msg varchar(255);  -- new trigger  if new.email like '%@hotmail.com' then  set err\_msg := concat('E-mail ', new.email, ' cannot be registered because it belongs to hotmail.com domain');  signal sqlstate '45000'  set message\_text = err\_msg;  … |

## Declare Inside

DECLARE commands are INSIDE begin… end

## Value Assignment

To assign value use

SET var1 := expression

## GET DIAGNOSTICS ( SQL%ROWCOUNT )

After successful statement processed row count can be retrieved like this:

|  |
| --- |
| /\* analog of Oracle SQL%rowcount \*/  set @nrows :=0;    delete from t1 where f1 != 'q';    GET DIAGNOSTICS @nrows = ROW\_COUNT;  select @nrows rows1;  INSERT INTO t1 (f1) VALUES('1adasdasd'), ('1sadasdasdasdasd');    -- Check whether the insert was successful  GET DIAGNOSTICS @nrows = ROW\_COUNT;  select @nrows rows2; |

## Exception handling

More info here:

<https://dev.mysql.com/doc/refman/8.0/en/declare-handler.html>

Exceptions are handled in declared handlers i.e.

|  |
| --- |
| DECLARE *handler\_action* HANDLER  FOR *condition\_value* [, *condition\_value*] ...  *statement*  *handler\_action*: {  CONTINUE  | EXIT  | UNDO  }  *condition\_value*: {  *mysql\_error\_code*  | SQLSTATE [VALUE] *sqlstate\_value*  | *condition\_name*  | SQLWARNING  | NOT FOUND  | SQLEXCEPTION  } |

### Get Error Message ( i.e. Sqlcode, sqlerrm)

Inside handler call GET DIAGNOSTICS CONDITION 1

|  |
| --- |
| BEGIN  -- Declare variables to hold diagnostics area information  DECLARE code CHAR(5) DEFAULT '00000';  DECLARE msg TEXT;  …  DECLARE CONTINUE  HANDLER FOR SQLEXCEPTION  BEGIN  GET DIAGNOSTICS CONDITION 1  code = RETURNED\_SQLSTATE, msg = MESSAGE\_TEXT;  END; |

### Continue Handler

DECLARE CONTINUE HANDLER FOR SQLEXCEPTION BEGIN .. END;

Process exception and continue execution

|  |
| --- |
| create procedure pec ( value int)  BEGIN  -- Declare variables to hold diagnostics area information  DECLARE code CHAR(5) DEFAULT '00000';  DECLARE msg TEXT;  DECLARE nrows INT;  DECLARE result TEXT;  -- Declare exception handler for failed insert  DECLARE CONTINUE  HANDLER FOR SQLEXCEPTION  BEGIN  GET DIAGNOSTICS CONDITION 1  code = RETURNED\_SQLSTATE, msg = MESSAGE\_TEXT;  END;  -- Perform the insert  INSERT INTO t1 (f1) VALUES(value);  -- Check whether the insert was successful  IF code = '00000' THEN  GET DIAGNOSTICS nrows = ROW\_COUNT;  SET result = CONCAT('insert succeeded, row count = ',nrows);  ELSE  SET result = CONCAT('insert failed, error = ',code,', message = ',msg);  END IF;  -- Say what happened  SELECT result;  END; |

### Exit Handler (i.e. EXCEPTION block )

Break normal execution and exit

|  |
| --- |
| create procedure pex ( value int)  BEGIN  -- Declare variables to hold diagnostics area information  DECLARE code CHAR(5) DEFAULT '00000';  DECLARE msg TEXT;  DECLARE nrows INT;  DECLARE result TEXT;  -- Declare exception handler for failed insert  DECLARE EXIT  HANDLER FOR SQLEXCEPTION  BEGIN  GET DIAGNOSTICS CONDITION 1  code = RETURNED\_SQLSTATE, msg = MESSAGE\_TEXT;  select concat( 'exiting with an error', msg) errmsg;  END;  -- Perform the insert  INSERT INTO t1 (f1) VALUES(value);  -- Check whether the insert was successful  IF code = '00000' THEN  GET DIAGNOSTICS nrows = ROW\_COUNT;  SET result = CONCAT('insert succeeded, row count = ',nrows);  ELSE  SET result = CONCAT('insert failed, error = ',code,', message = ',msg);  END IF;  -- Say what happened  SELECT result;  END; |

## Triggers

This is an example of Before Insert trigger that raises custom error

|  |
| --- |
| delimiter //  drop trigger if exists trg\_users\_bi  //  create trigger trg\_users\_bi  before insert on join\_us.users for each row  begin  declare err\_msg varchar(255);  -- new trigger  if new.email like '%@hotmail.com' then  set err\_msg := concat('E-mail ', new.email, ' cannot be registered because it belongs to hotmail.com domain');  signal sqlstate '45000'  set message\_text = err\_msg;  end if;  end;  //  delimiter ; |

## Functions

## Procedures