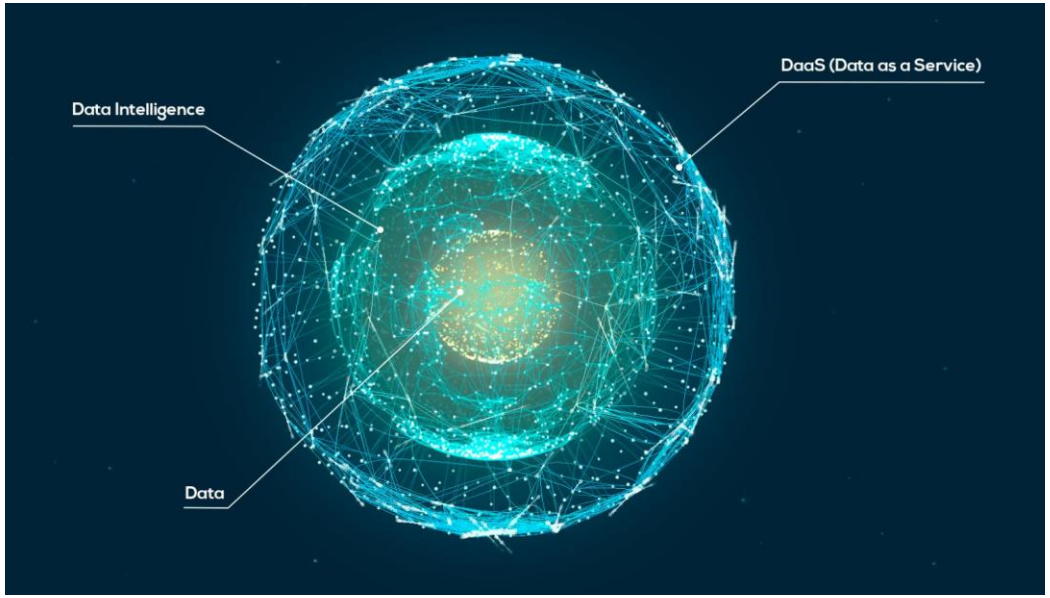
Application Design Using Java

Lecture 21

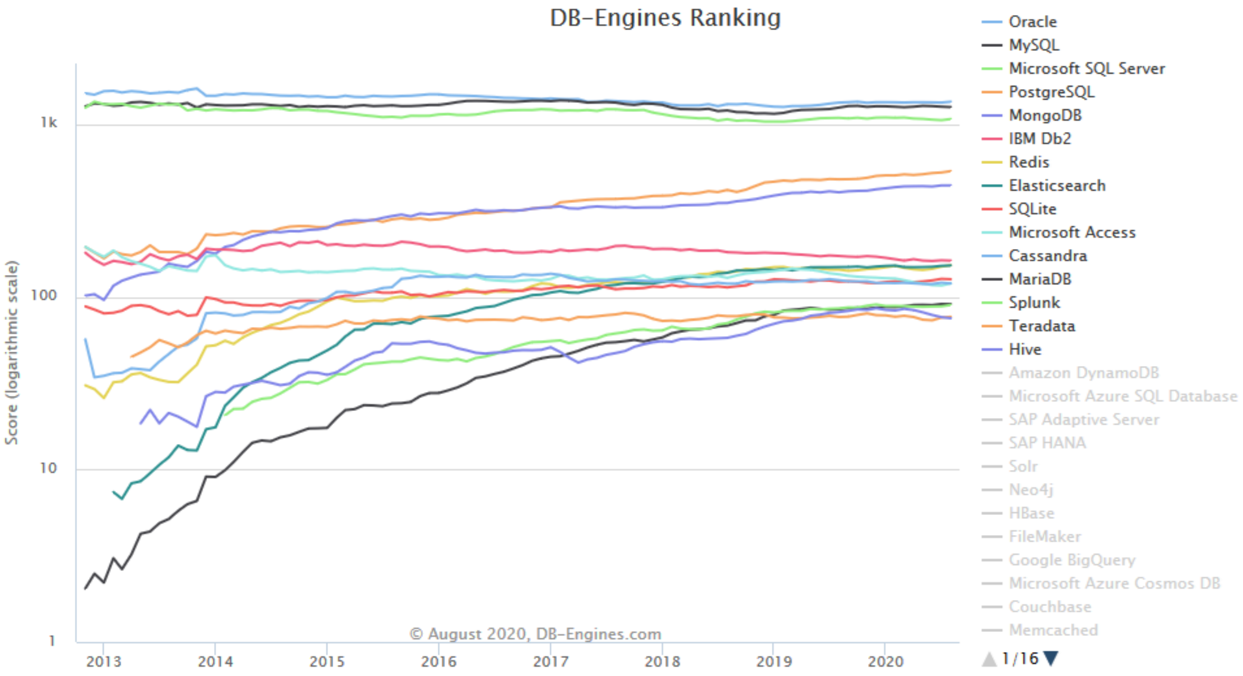
Data-centric Applications

- Data is the primary and permanent asset
- Applications come and go
- Data model precedes the implementation of any given application and will be around and valid long after it is gone



MySQL RDBMS

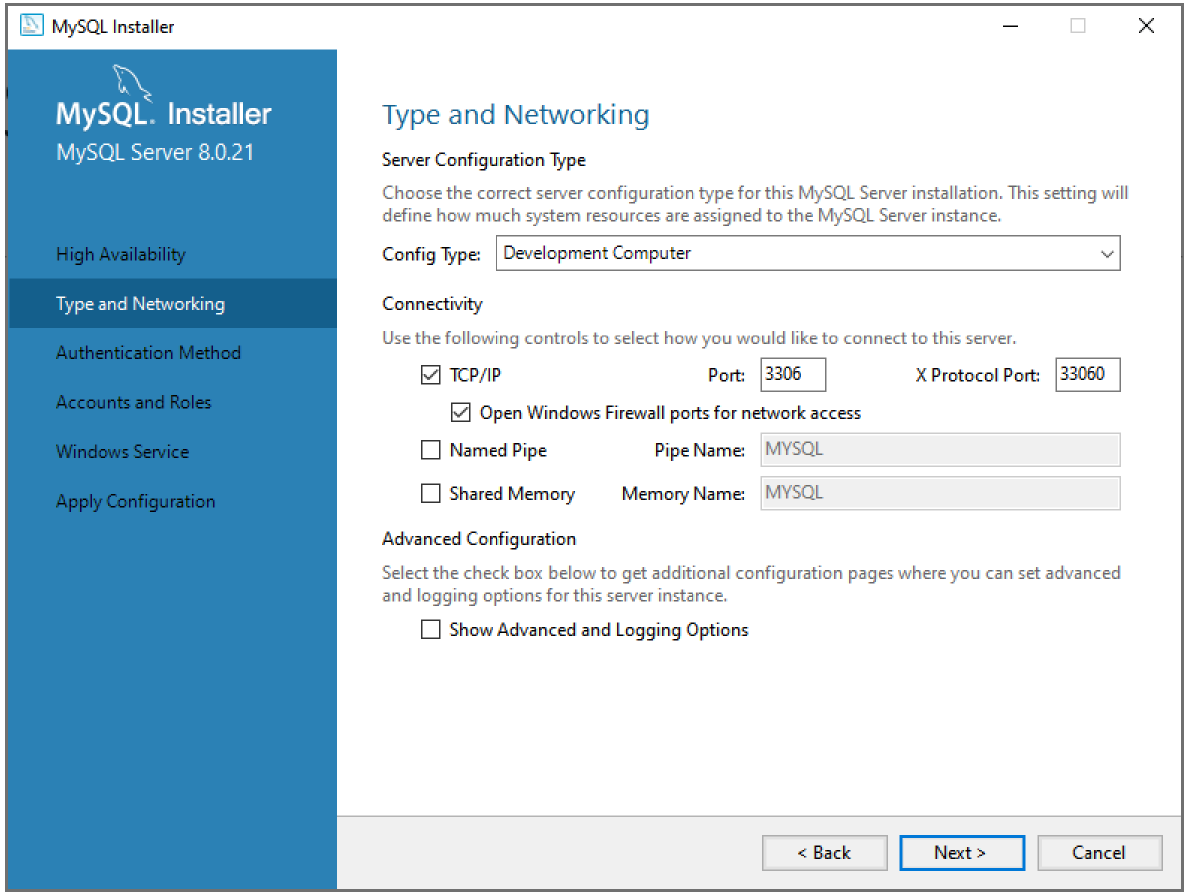
- Works on almost all modern platforms
- Different editions, including freely downloadable Community Edition
- One of the most popular RDBMS'*
- No internal limits on the size of the tables
- Open source
- Acquired by Sun Microsystems in 2008 which was then acquired by Oracle in 2010
- Original developers of MySQL created a MySQL fork called MariaDB
  - Guaranteed to stay open source
  - Default in most Linux distributions



* According to https://db-engines.com

MySQL Configuration I

- Note the TCP/IP port number



MySQL Installer

MySQL. Installer
MySQL Server 8.0.21

Type and Networking

Server Configuration Type

Choose the correct server configuration type for this MySQL Server installation. This setting will define how much system resources are assigned to the MySQL Server instance.

Config Type: Development Computer

Connectivity

Use the following controls to select how you would like to connect to this server.

☑ TCP/IP   Port: 3306   X Protocol Port: 33060

☑ Open Windows Firewall ports for network access

☐ Named Pipe   Pipe Name: MYSQL

☐ Shared Memory   Memory Name: MYSQL

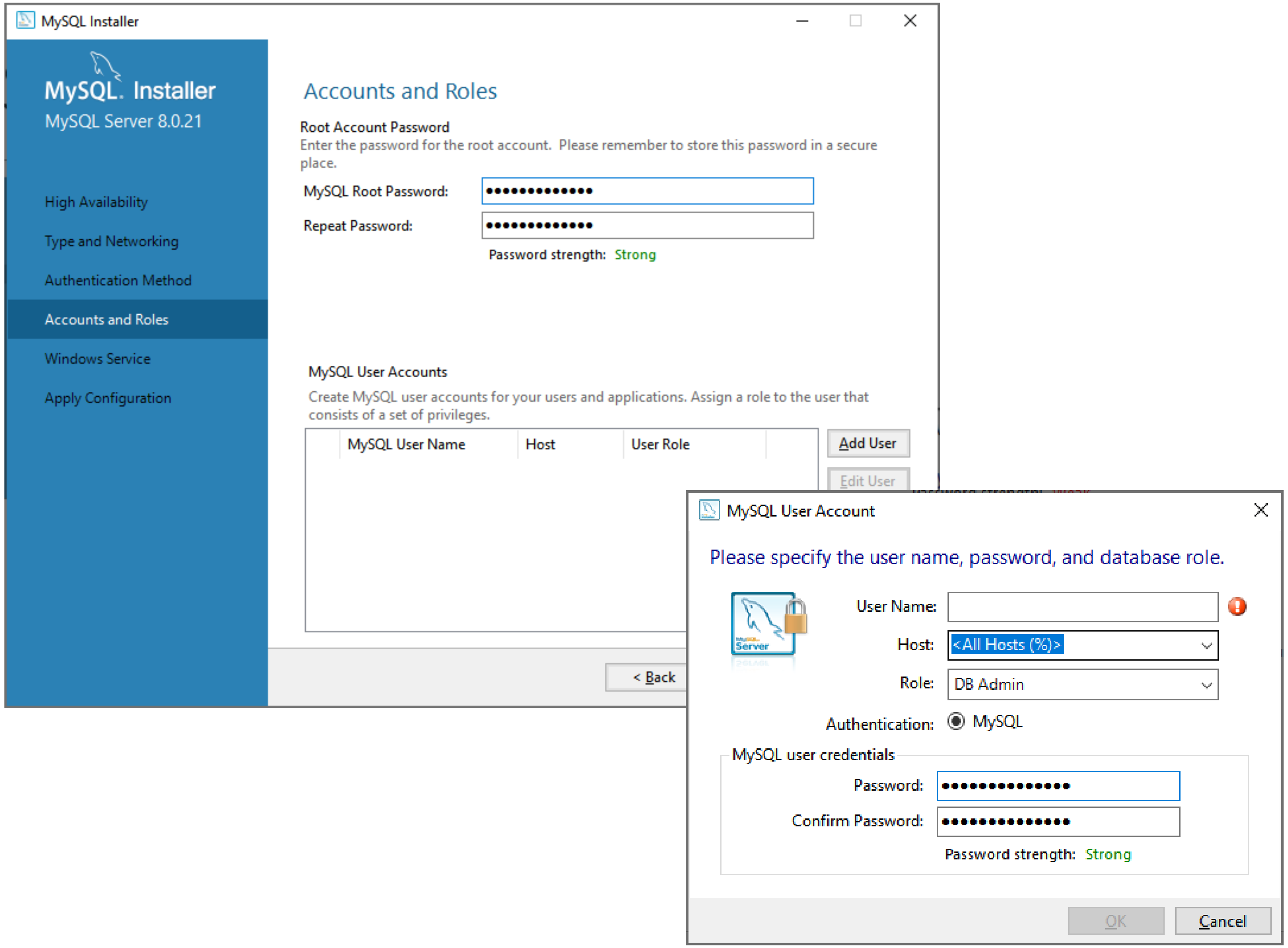
Advanced Configuration

Select the check box below to get additional configuration pages where you can set advanced and logging options for this server instance.

☐ Show Advanced and Logging Options

MySQL Configuration II

- Make sure to select a strong Root password
- Create at least one user with the Role of DB Admin
- You may create additional users with different roles



Accounts and Roles

Root Account Password

Enter the password for the root account. Please remember to store this password in a secure place.

MySQL Root Password: ••••••••••••••

Repeat Password: ••••••••••••••

Password strength: Strong

MySQL User Accounts

Create MySQL user accounts for your users and applications. Assign a role to the user that consists of a set of privileges.

MySQL User Name	Host	User Role

MySQL User Account

Please specify the user name, password, and database role.

User Name:

Host: <All Hosts (%)>

Role: DB Admin

Authentication: ● MySQL

MySQL user credentials

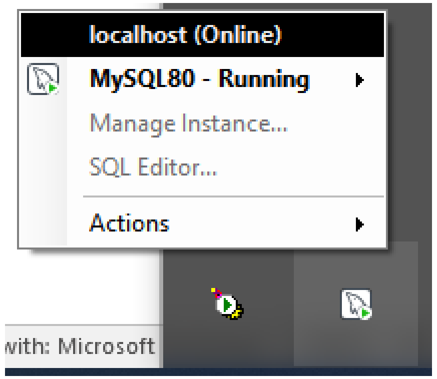
Password: •••••••••••••••

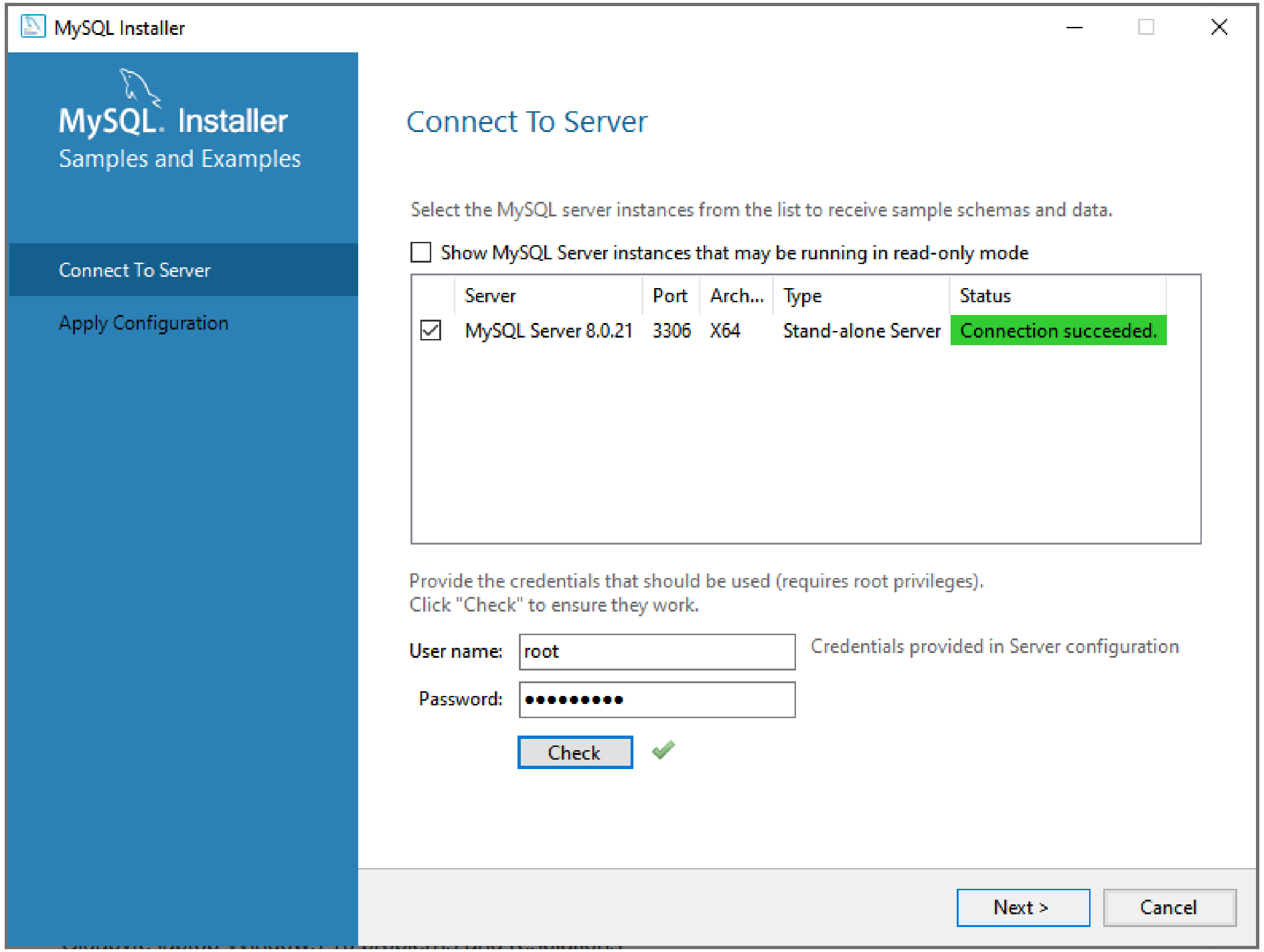
Confirm Password: •••••••••••••••

Password strength: Strong

MySQL Configuration III

- Install samples and examples





Connect To Server

Select the MySQL server instances from the list to receive sample schemas and data.

☐ Show MySQL Server instances that may be running in read-only mode

	Server	Port	Arch...	Type	Status
☑	MySQL Server 8.0.21	3306	X64	Stand-alone Server	Connection succeeded.

Provide the credentials that should be used (requires root privileges).
Click "Check" to ensure they work.

User name: root

Password: •••••••••

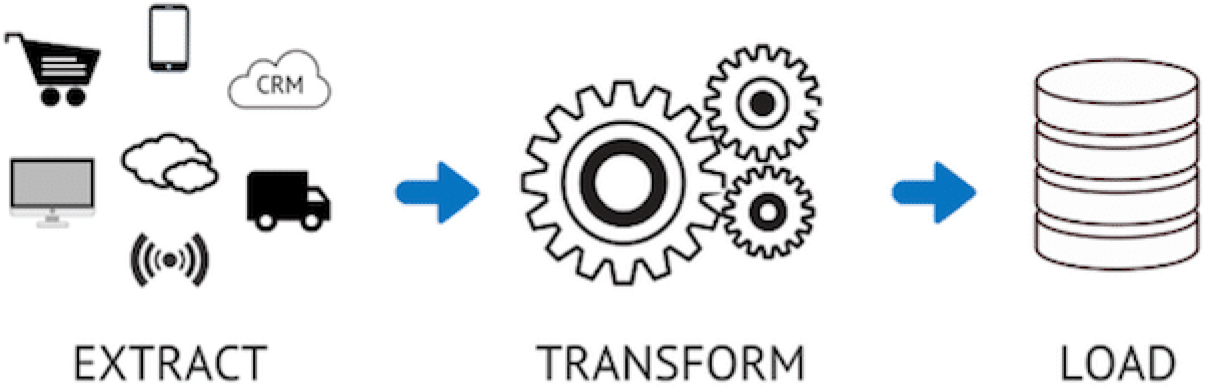
Credentials provided in Server configuration

Check ✓

Extract, Transform, and Load (ETL)

CSCI-4350 Data Science

- Data integration
- Allows different data to work together
- Allows consolidation of data from multiple sources into a single centralized location



Extract

- Extract raw data from different sources
- Possible sources
  - Existing databases and legacy systems
  - Cloud, hybrid, and on-premises environments
  - Sales and marketing applications
  - Mobile devices and apps
  - CRM systems
  - Data storage platforms
  - Data warehouses
  - Analytics tools
  - Web sites

Transform

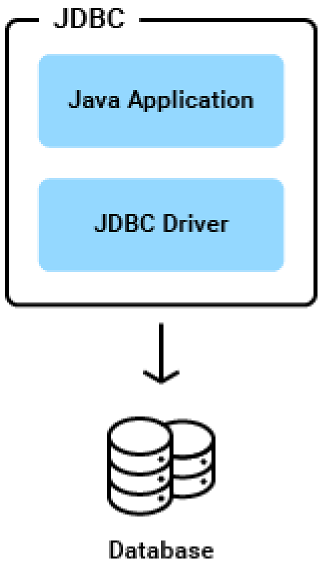
- Apply rules and regulations to ensure data quality and accessibility
- Processes
  - Cleansing — inconsistencies and missing values in the data are resolved
  - Standardization — formatting rule are applied to the data set
  - Deduplication — redundant data is excluded or discarded
  - Verification — unusable data is removed and anomalies are flagged
  - Sorting — data is organized according to type
  - Other tasks — any additional/optional rules can be applied to improve data quality

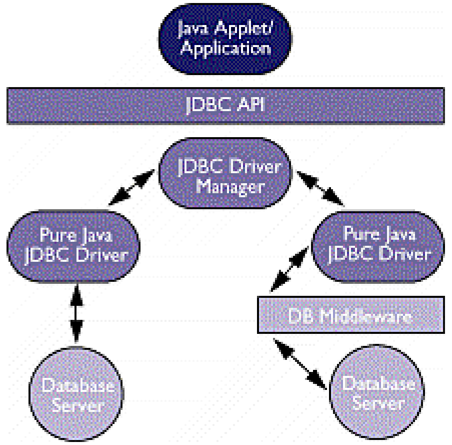
Load

- Load the newly transformed data into a new destination
- Loading methods
  - Full loading — data loaded at once
  - Incremental loading — data loaded at scheduled intervals

Java Database Connectivity (JDBC)

- An application programming interface (API) which defines how a client may access a database
- Unified approach to accessing data on any supported RDBMS
- Requires the “driver”, e.g., mysql-connector-java-8.0.11.jar
- Uses a *connection string* to specify parameters of the connection:





JDBC Drivers

- Type 1
  - Translates JDBC to ODBC
  - Relies on an ODBC driver to communicate with the database
- Type 2
  - Written partly in Java and partly in native code
  - Communicates with the client API of a database
  - When using such a driver, you must install some platform-specific code onto the client in addition to a Java library
- Type 3
  - A pure Java client library
  - Uses a database-independent protocol to communicate database requests to a server component, which then translates the requests into a database-specific protocol
  - This simplifies deployment because the platform-specific code is located only on the server
- Type 4
  - A pure Java library
  - Translates JDBC requests directly to a database-specific protocol

MySQL Connection String

```
protocol//[hosts][/database][?properties]
```

- `protocol` — possible protocols for a connection
- `hosts` — host names and user credentials
- `database` — the default database or catalog to open
- `properties` — a succession of global properties applying to all hosts, preceded by ? and written as key=value pairs separated by the symbol "&"

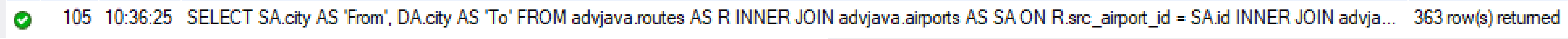
Example connection string: `jdbc:mysql://(host=myhost1,port=1111, user=sandy,password=secret)/db?key1=value1&key2=value2&key3=value3`

Types of Queries

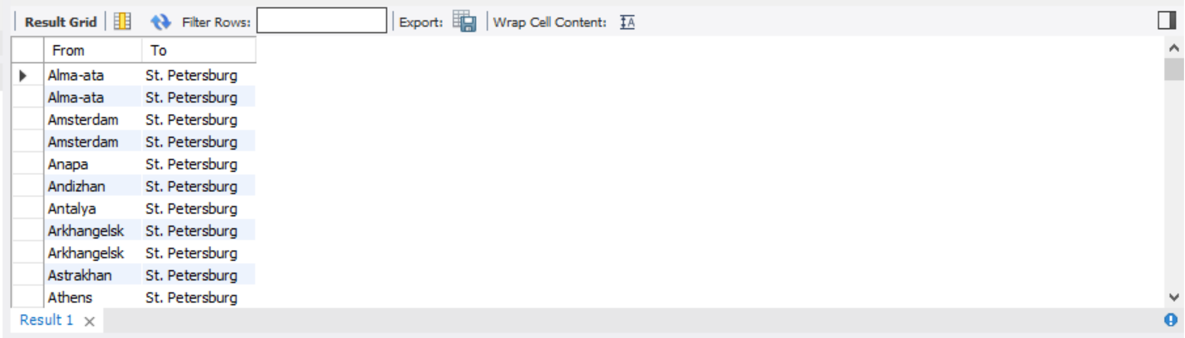
- Returns a result set
- Returns the number of rows affected
- Returns status code

Result set

- SELECT SA.city AS 'From', DA.city AS 'To'FROM advjava.routes AS RINNER JOIN advjava.airports AS SA ON R.src_airport_id = SA.idINNER JOIN advjava.airports AS DA ON R.dst_airport_id = DA.idWHERE SA.iata = 'LED' OR DA.iata = 'LED'ORDER BY SA.city ASC



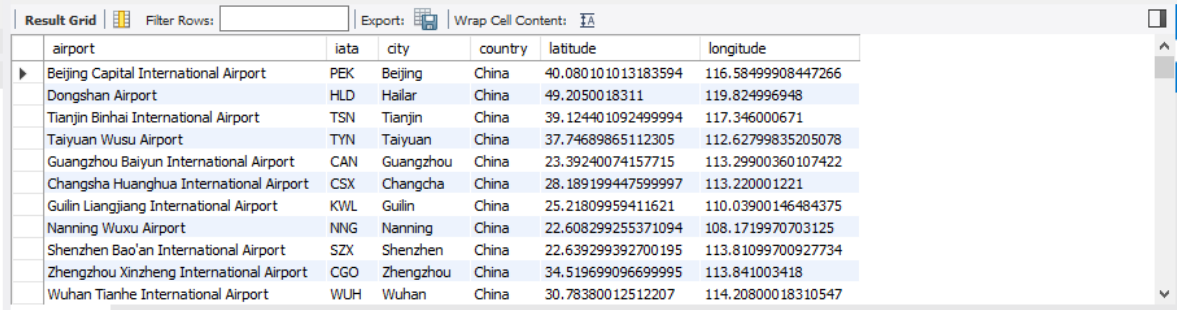
```
105  10:36:25  SELECT SA.city AS 'From', DA.city AS 'To' FROM advjava.routes AS R INNER JOIN advjava.airports AS SA ON R.src_airport_id = SA.id INNER JOIN advja...   363 row(s) returned
```



From	To
Alma-ata	St. Petersburg
Alma-ata	St. Petersburg
Amsterdam	St. Petersburg
Amsterdam	St. Petersburg
Anapa	St. Petersburg
Andizhan	St. Petersburg
Antalya	St. Petersburg
Arkhangelsk	St. Petersburg
Arkhangelsk	St. Petersburg
Astrakhan	St. Petersburg
Athens	St. Petersburg

- CALL \`advjava\`.\`airports_by_country\`('China');

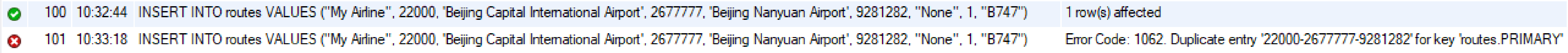
```
111  10:38:02  CALL `advjava`.`airports_by_country`('China')                                   171 row(s) returned
```



airport	iata	city	country	latitude	longitude
Beijing Capital International Airport	PEK	Beijing	China	40.080101013183594	116.58499908447266
Dongshan Airport	HLD	Hailar	China	49.2050018311	119.824996948
Tianjin Binhai International Airport	TSN	Tianjin	China	39.124401092499994	117.346000671
Taiyuan Wusu Airport	TYN	Taiyuan	China	37.74689865112305	112.62799835205078
Guangzhou Baiyun International Airport	CAN	Guangzhou	China	23.39240074157715	113.29900360107422
Changsha Huanghua International Airport	CSX	Changcha	China	28.189199447599997	113.220001221
Guilin Liangjiang International Airport	KWL	Guilin	China	25.21809959411621	110.03900146484375
Nanning Wuxu Airport	NNG	Nanning	China	22.608299255371094	108.1719970703125
Shenzhen Bao'an International Airport	SZX	Shenzhen	China	22.639299392700195	113.81099700927734
Zhengzhou Xinzheng International Airport	CGO	Zhengzhou	China	34.519699096699995	113.841003418
Wuhan Tianhe International Airport	WUH	Wuhan	China	30.78380012512207	114.20800018310547

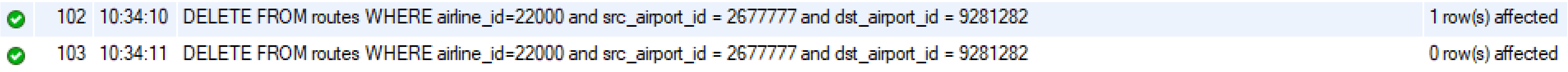
Rows affected

- `INSERT INTO routes VALUES ("My Airline", 22000, 'Beijing Capital International Airport',2677777, 'Beijing Nanyuan Airport', 9281282, "None", 1, "B747");`



```
✓  100  10:32:44  INSERT INTO routes VALUES ("My Airline", 22000, 'Beijing Capital International Airport', 2677777, 'Beijing Nanyuan Airport', 9281282, "None", 1, "B747")  1 row(s) affected
✗  101  10:33:18  INSERT INTO routes VALUES ("My Airline", 22000, 'Beijing Capital International Airport', 2677777, 'Beijing Nanyuan Airport', 9281282, "None", 1, "B747")  Error Code: 1062. Duplicate entry '22000-2677777-9281282' for key 'routes.PRIMARY'
```

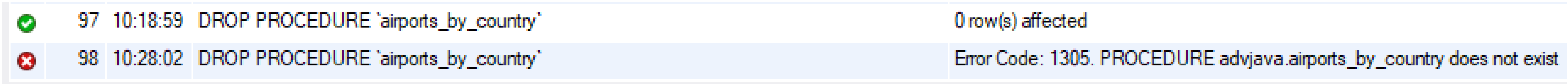
- `DELETE FROM routes WHERE airline_id=22000 and src_airport_id = 2677777 and dst_airport_id = 9281282;`



```
✓  102  10:34:10  DELETE FROM routes WHERE airline_id=22000 and src_airport_id = 2677777 and dst_airport_id = 9281282  1 row(s) affected
✓  103  10:34:11  DELETE FROM routes WHERE airline_id=22000 and src_airport_id = 2677777 and dst_airport_id = 9281282  0 row(s) affected
```

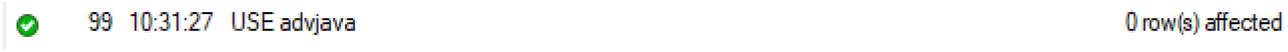
Status Code

- `DROP PROCEDURE `airports_by_country`;`



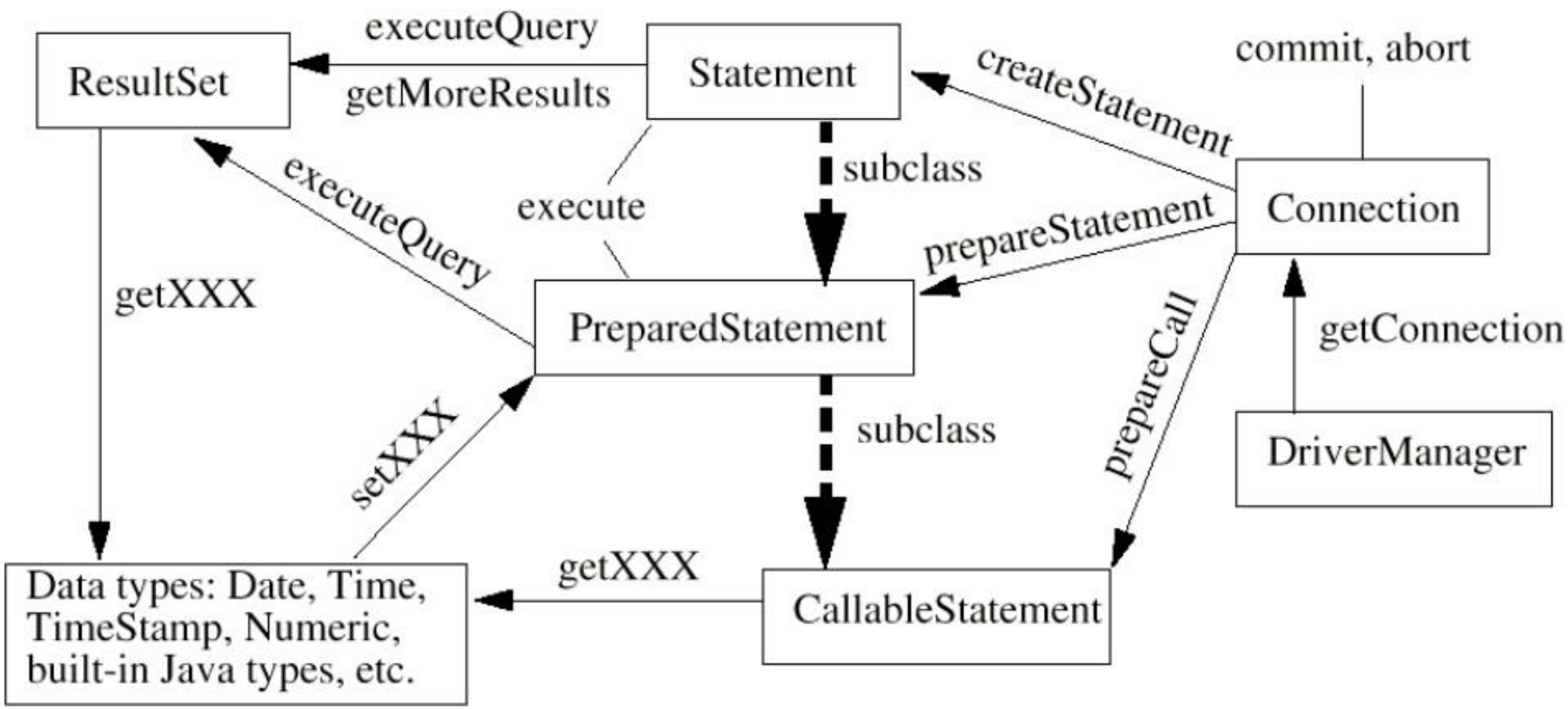
```
✓   97  10:18:59  DROP PROCEDURE `airports_by_country`    0 row(s) affected
✗   98  10:28:02  DROP PROCEDURE `airports_by_country`    Error Code: 1305. PROCEDURE advjava.airports_by_country does not exist
```

- `USE advjava;`



```
✓   99  10:31:27  USE advjava                             0 row(s) affected
```

JDBC Class Diagram



Stored Procedures I

- Prepared SQL code
- Stored in the database
- Can be reused over and over again
- Can have parameters that make the stored procedure more flexible and useful
- Repeated invocations of the same stored procedure execute from the cache without having to recompile it
- May contain control flow statements such as IF, CASE, and LOOP that allow you to implement the code in the procedural way
- Can call other stored procedures or stored functions promoting code modularization

Stored Procedures II

- Advantages
  - Reduce network traffic
  - Centralize business logic in the database
  - Enhance database security
- Disadvantages
  - Database server memory usage can be higher
  - Harder to debug than SQL statements
  - Require a specialized skill set to develop and maintain

Input Parameters

- `IN` keyword
- This is the default mode
- The caller has to pass the value
- Protected from changes inside the stored procedure
    - Stored procedure only works on the copy of the IN parameter
    - Original value is retained after the stored procedure ends

```
DELIMITER //

CREATE PROCEDURE GetOfficeByCountry(
        IN countryName VARCHAR(255)
)
BEGIN
        SELECT *
        FROM offices
        WHERE country = countryName;
END //

DELIMITER ;
```

```
CALL GetOfficeByCountry('USA');
```

Output Parameters

- `OUT` keyword
- Must be a variable, not a constant
- Can be changed inside the stored procedure
- New value is passed back to the caller
- Stored procedure cannot access the initial value of the OUT parameter when it starts

```
DELIMITER $$

CREATE PROCEDURE GetOrderCountByStatus (
        IN  orderStatus VARCHAR(25),
        OUT total INT
)
BEGIN
        SELECT COUNT(orderNumber)
        INTO total
        FROM orders
        WHERE status = orderStatus;
END$$

DELIMITER ;
```

```
CALL GetOrderCountByStatus('Shipped',@total);
SELECT @total;
```

Input/Output Parameters

- `INOUT` keyword
- A combination of `IN` and `OUT` parameters
- Caller may pass the argument
- Stored procedure can modify the `INOUT` parameter and pass the new value back to the caller

```
DELIMITER $$

CREATE PROCEDURE SetCounter(
        INOUT counter INT,
    IN inc INT
)
BEGIN
        SET counter = counter + inc;
END$$

DELIMITER ;
```

```
SET @counter = 1;
CALL SetCounter(@counter,1); -- 2
CALL SetCounter(@counter,1); -- 3
CALL SetCounter(@counter,5); -- 8
SELECT @counter; -- 8
```

//TODO before next lecture:

- Homework 4 due on 4/16 at 11:59 pm EDT. Must be submitted on Submitty.