

Chapter 1

An introduction to relational databases and SQL

Objectives

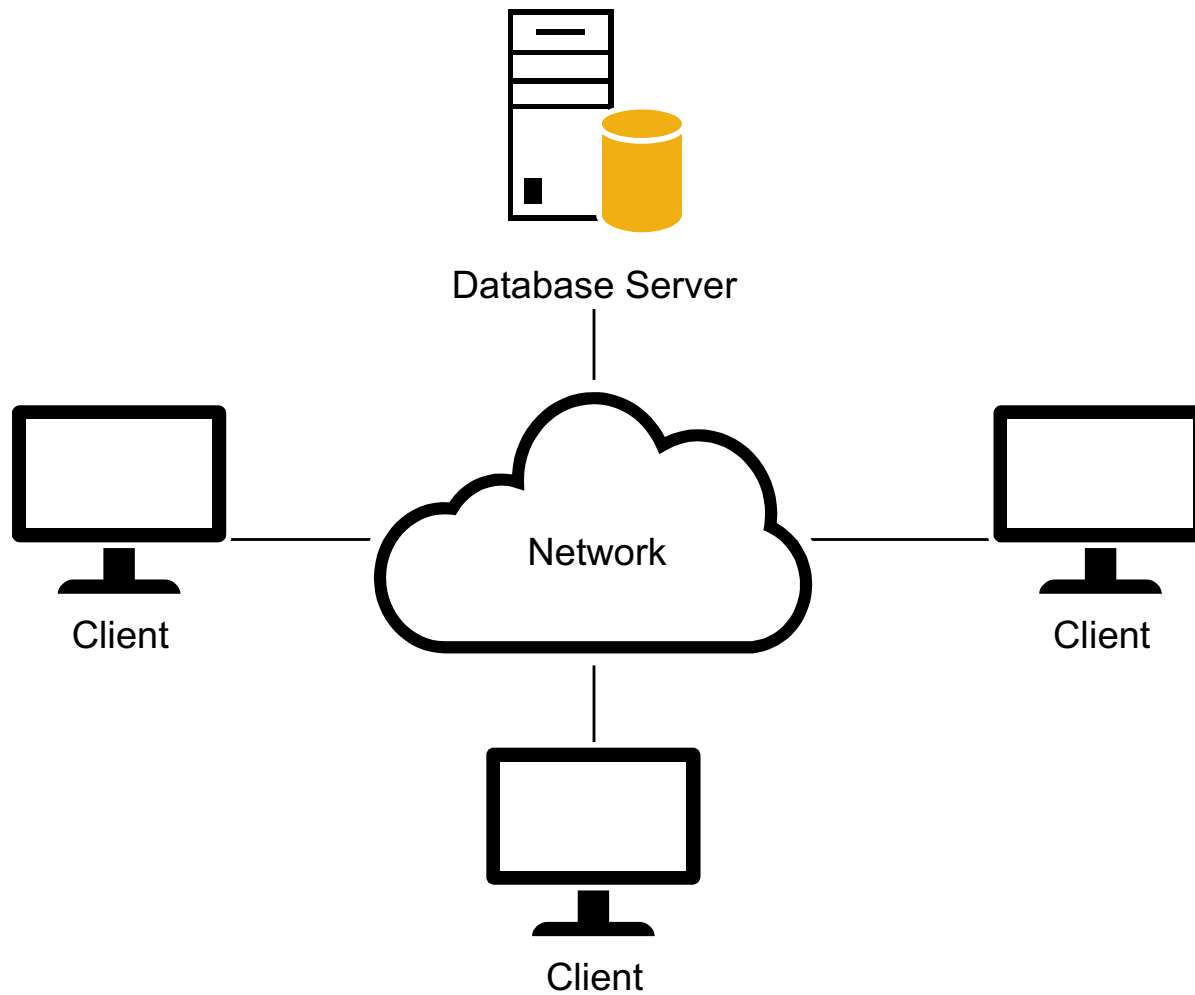
Knowledge

- Identify the three main hardware components of a client/server system.
- Describe the way a client accesses the database on a server using these terms: application software, data access API, database management system, SQL query, and query results.
- Describe the way a relational database is organized using these terms: tables, columns, rows, cells, primary keys, and foreign keys.
- Identify the three types of relationships that can exist between two tables.
- Describe the way the columns in a table are defined using these terms: data type, null value, and default value.

Objective (cont.)

- Describe how an entity relationship diagram can show how the tables in a database are defined and related.
- Describe the difference between DML statements and DDL statements.
- List three coding techniques that can make your SQL code easier to read and maintain.
- Describe the use of a database driver.

A simple client/server system



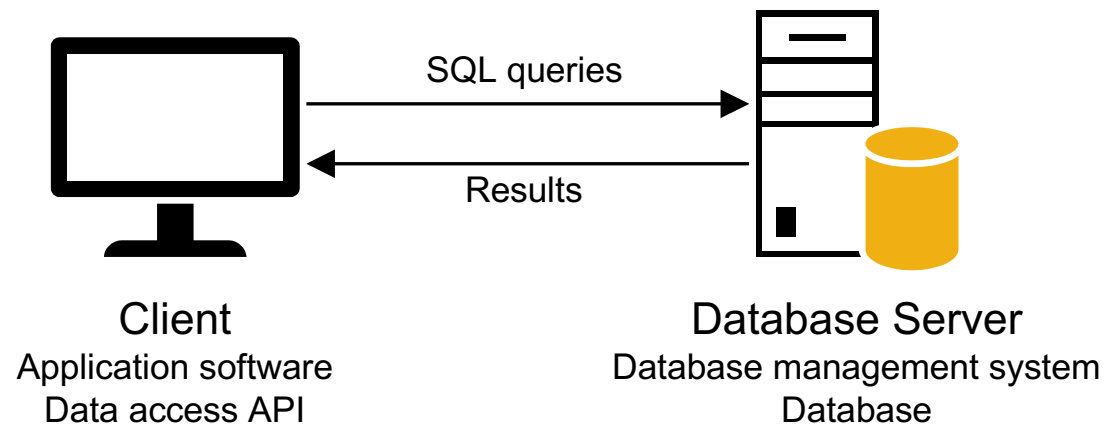
The three hardware components of a client/server system

- Clients
- Server
- Network

Terms to know

- Local area network (LAN)
- Wide area network (WAN)
- Enterprise system

Client software, server software, and the SQL interface



Server software

- Database management system (DBMS)
- The DBMS does the *back-end processing*

Client software

- Application software
- Data access API (application programming interface)
- The client software does the *front-end processing*

The SQL interface

- The application software communicates with the DBMS by sending SQL queries through the data access API.
- When the DBMS receives a query, it provides a service like returning the requested data (the query results) to the client.
- *SQL* stands for *Structured Query Language*, which is the standard language for working with a relational database.

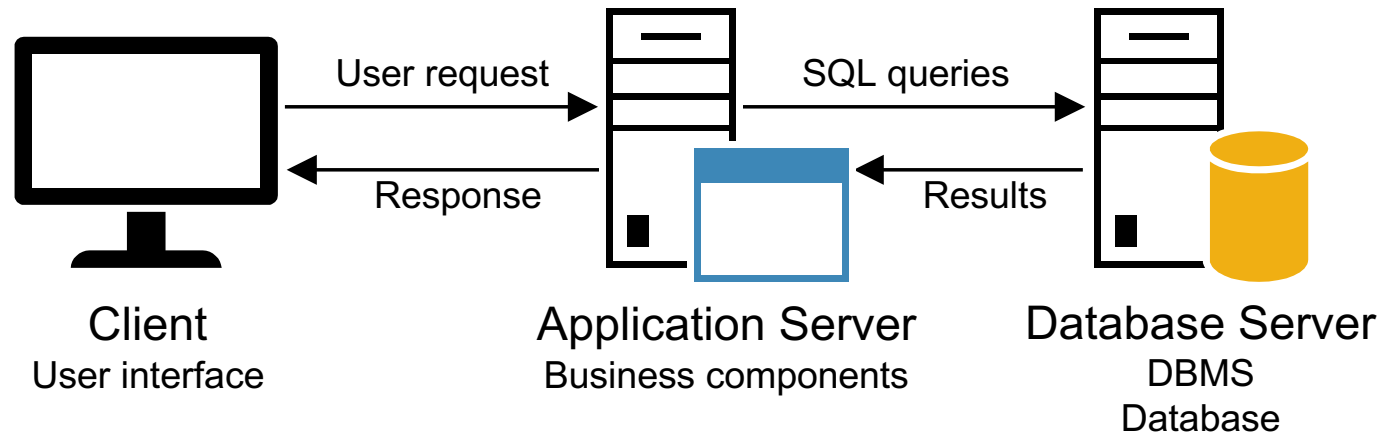
Client/server system

- Processing is divided between client and server.

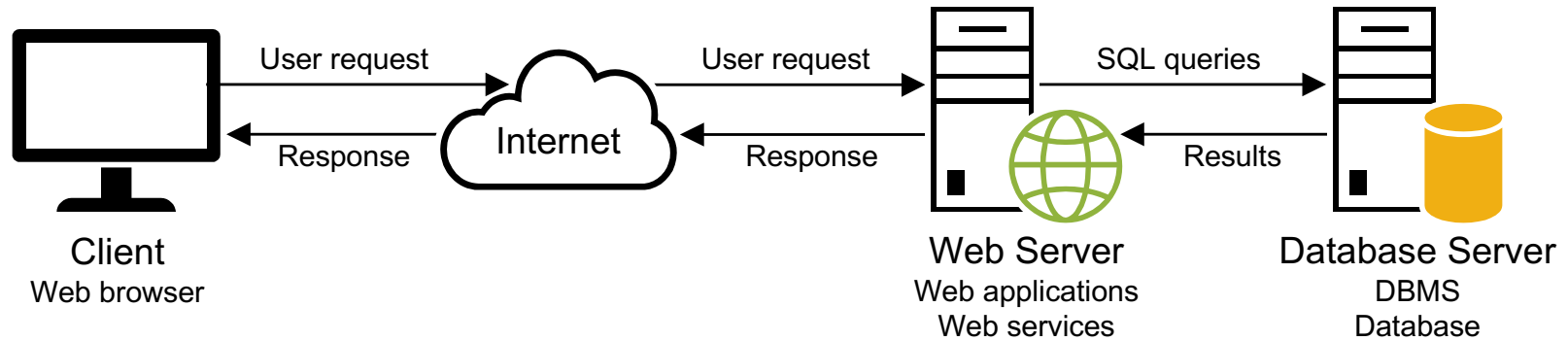
File-handling system

- All processing is done on the clients.

A networked system that uses an application server



A simple web-based system



Other client/server components

- Application servers store business components
- Web servers store web applications and web services

How web applications work

- A web browser on a client sends a request to a web server.
- The web server processes the request.
- The web server passes any data requests to the database server.
- The database server returns results to web server.
- The web server returns a response to the browser.

The Vendors table in an Accounts Payable (AP) database

Primary key **Columns**

vendor_id	vendor_name	vendor_address1	vendor_address2	vendor_city	vendor_st
1	US Postal Service	Attn: Supt. Window Services	PO Box 7005	Madison	WI
2	National Information Data Ctr	PO Box 96621	NULL	Washington	DC
3	Register of Copyrights	Library Of Congress	NULL	Washington	DC
4	Jobtrak	1990 Westwood Blvd Ste 260	NULL	Los Angeles	CA
5	Newbrige Book Clubs	3000 Cindel Drive	NULL	Washington	NJ
6	California Chamber Of Commerce	3255 Ramos Cir	NULL	Sacramento	CA
7	Towne Advertiser's Mailing Svcs	Kevin Minder	3441 W Macarthur Blvd	Santa Ana	CA
8	BFI Industries	PO Box 9369	NULL	Fresno	CA
9	Pacific Gas & Electric	Box 52001	NULL	San Francisco	CA
10	Robbins Mobile Lock And Key	4669 N Fresno	NULL	Fresno	CA
11	Bill Marvin Electric Inc	4583 E Home	NULL	Fresno	CA
12	City Of Fresno	PO Box 2069	NULL	Fresno	CA

Rows

Terms to know

- Relational database
- Table
- Column
- Row
- Cell
- Primary key
- Composite primary key
- Non-primary key (unique key)
- Index

The relationship between two tables

Primary key

vendor_id	vendor_name	vendor_address1	vendor_address2	vendor_city	vendor_state
114	Postmaster	Postage Due Technician	1900 E Street	Fresno	CA
115	Roadway Package System, Inc	Dept La 21095	NULL	Pasadena	CA
116	State of California	Employment Development ...	PO Box 826276	Sacramento	CA
117	Suburban Propane	2874 S Cherry Ave	NULL	Fresno	CA
118	Unocal	P.O. Box 860070	NULL	Pasadena	CA
119	Yesmed, Inc	PO Box 2061	NULL	Fresno	CA
120	Dataforms/West	1617 W. Shaw Avenue	Suite F	Fresno	CA
121	Zylka Design	3467 W Shaw Ave #103	NULL	Fresno	CA
122	United Parcel Service	P.O. Box 505820	NULL	Reno	NV
123	Federal Express Corporation	P.O. Box 1140	Dept A	Memphis	TN

invoice_id	vendor_id	invoice_number	invoice_date	invoice_total	payment_total	credit_total
55	123	963253245	2014-06-10	40.75	40.75	0.00
56	86	367447	2014-06-11	2433.00	2433.00	0.00
57	103	75C-90227	2014-06-11	1367.50	1367.50	0.00
58	123	963253256	2014-06-11	53.25	53.25	0.00
59	123	4-314-3057	2014-06-11	13.75	13.75	0.00
60	122	989319-497	2014-06-12	2312.20	2312.20	0.00
61	115	24946731	2014-06-15	25.67	25.67	0.00
62	123	963253269	2014-06-15	26.75	26.75	0.00
63	122	989319-427	2014-06-16	2115.81	2115.81	0.00
64	123	963253267	2014-06-17	23.50	23.50	0.00

Foreign key

Terms to know

- Foreign key
- Referential integrity
- One-to-many relationship
- One-to-one relationship
- Many-to-many relationship

The columns of the Invoices table

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
invoice_id	INT(11)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
vendor_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
invoice_number	VARCHAR(50)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
invoice_date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
invoice_total	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
payment_total	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0.00'
credit_total	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	'0.00'
terms_id	INT(11)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
invoice_due_date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
payment_date	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Column Name:	<input type="text"/>	Data Type:	<input type="text"/>
Collation:	<input type="text"/>	Default:	<input type="text"/>
Comments:	<input type="text"/>		
	<input type="checkbox"/> Primary	<input type="checkbox"/> Not Null	<input type="checkbox"/> Unique
	<input type="checkbox"/> Binary	<input type="checkbox"/> Unsigned	<input type="checkbox"/> Zero Fill
	<input type="checkbox"/> Auto Increment		

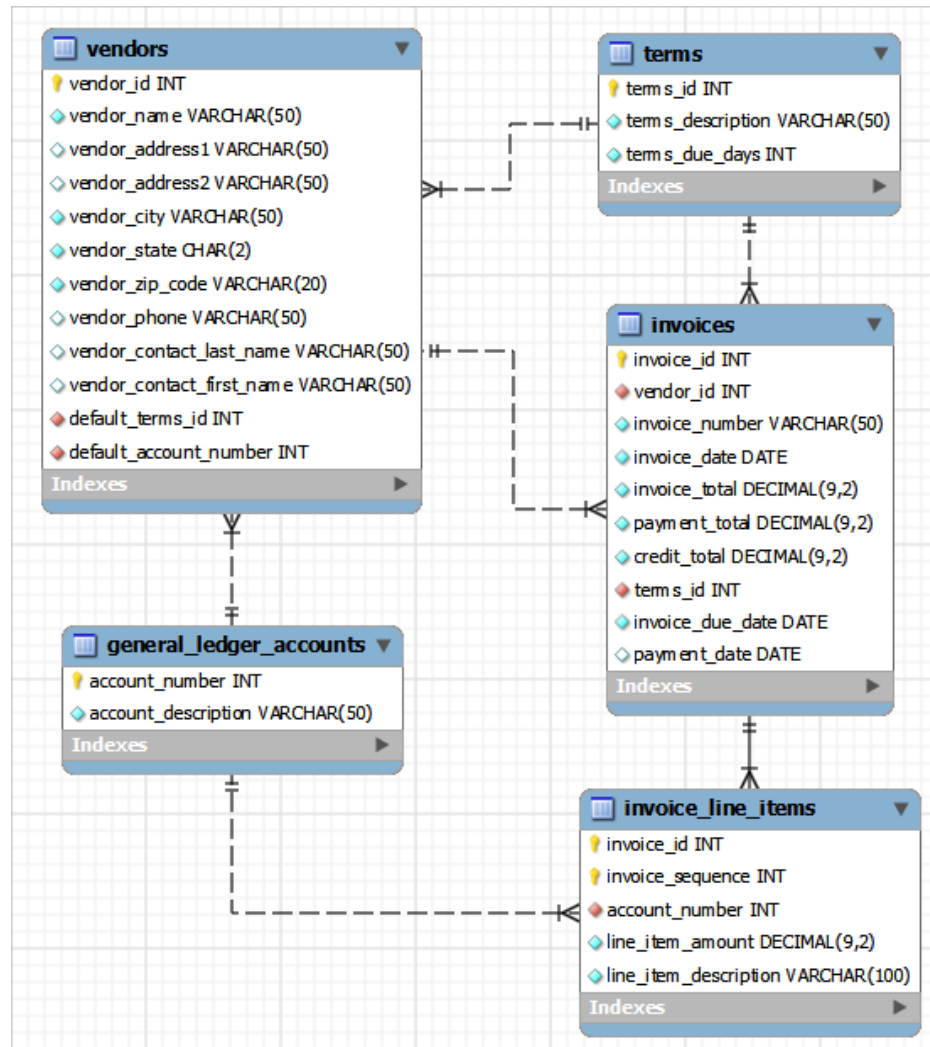
Common MySQL data types

- CHAR, VARCHAR
- INT, DECIMAL
- FLOAT
- DATE

Terms to know

- Data type
- Null value
- Default value
- Auto increment column

An EER diagram for the AP database



Important events in the history of SQL

Year	Event
1970	Dr. E. F. Codd developed the relational database model.
1979	Relational Software, Inc. (later renamed Oracle) released the first relational DBMS, Oracle.
1982	IBM released their first RDBMS, SQL/DS (SQL/Data System).
1985	IBM released DB2 (Database 2).
1987	Microsoft released SQL Server.
1989	ANSI published the first set of standards (ANSI/ISO SQL-89, or SQL1).
1992	ANSI revised standards (ANSI/ISO SQL-92, or SQL2)
1995	MySQL AB released MySQL for internal use.
1999	ANSI published SQL3 (ANSI/ISO SQL:1999).

Important events in the history of SQL (continued)

Year	Event
2000	MySQL became an open-source database.
2003	ANSI published SQL4 (ANSI/ISO SQL:2003).
2008	Sun Microsystems acquired MySQL.
2010	Oracle acquired Sun Microsystems and MySQL.

How knowing “standard SQL” helps you

- Basic SQL statements are the same for all *dialects*.
- Once you know one dialect, you can easily learn others.

How knowing “standard SQL” does not help you

- Most applications require modification when moved to another database.

First database releases

Oracle	1979
DB2	1985
SQL Server	1987
MySQL	2000

Primary platforms

Oracle	Unix, Windows, Mac OS OS/390 and z/OS
DB2	OS/390 and z/OS Unix, Windows, Mac OS
SQL Server	Windows
MySQL	Unix, z/OS Windows Mac OS

SQL DML statements

- SELECT
- INSERT
- UPDATE
- DELETE

SQL DDL statements

- CREATE DATABASE, TABLE, INDEX
- ALTER TABLE, INDEX
- DROP DATABASE, TABLE, INDEX

A statement that creates a new database

```
CREATE DATABASE ap
```

A statement that selects the current database

```
USE ap
```

A statement that creates a new table

```
CREATE TABLE invoices
(
  invoice_id          INT          PRIMARY KEY
                    AUTO_INCREMENT,
  vendor_id          INT          NOT NULL,
  invoice_number     VARCHAR(50)  NOT NULL,
  invoice_date       DATE         NOT NULL,
  invoice_total      DECIMAL(9,2) NOT NULL,
  payment_total      DECIMAL(9,2)          DEFAULT 0,
  credit_total       DECIMAL(9,2)          DEFAULT 0,
  terms_id           INT          NOT NULL,
  invoice_due_date   DATE         NOT NULL,
  payment_date       DATE,
  CONSTRAINT invoices_fk_vendors
    FOREIGN KEY (vendor_id)
    REFERENCES vendors (vendor_id),
  CONSTRAINT invoices_fk_terms
    FOREIGN KEY (terms_id)
    REFERENCES terms (terms_id)
)
```

A statement that adds a new column to a table

```
ALTER TABLE invoices  
ADD balance_due DECIMAL(9,2)
```

A statement that deletes the new column

```
ALTER TABLE invoices  
DROP COLUMN balance_due
```

A statement that creates an index on the table

```
CREATE INDEX invoices_vendor_id_index  
ON invoices (vendor_id)
```

A statement that deletes the new index

```
DROP INDEX invoices_vendor_id_index
```

The Invoices base table

invoice_id	vendor_id	invoice_number	invoice_date	invoice_total	payment_total	credit_total	terms_id	invoice_date
1	122	989319-457	2014-04-08	3813.33	3813.33	0.00	3	2014-05-08
2	123	263253241	2014-04-10	40.20	40.20	0.00	3	2014-05-10
3	123	963253234	2014-04-13	138.75	138.75	0.00	3	2014-05-13
4	123	2-000-2993	2014-04-16	144.70	144.70	0.00	3	2014-05-16
5	123	963253251	2014-04-16	15.50	15.50	0.00	3	2014-05-16

A SELECT statement that retrieves and sorts selected columns and rows

```
SELECT invoice_number, invoice_date, invoice_total,  
       payment_total, credit_total,  
       invoice_total - payment_total - credit_total  
       AS balance_due  
FROM invoices  
WHERE invoice_total - payment_total - credit_total > 0  
ORDER BY invoice_date
```

The result set defined by the SELECT statement

	invoice_number	invoice_date	invoice_total	payment_total	credit_total	balance_due	
▶	39104	2014-07-10	85.31	0.00	0.00	85.31	▲
	963253264	2014-07-18	52.25	0.00	0.00	52.25	☰
	31361833	2014-07-21	579.42	0.00	0.00	579.42	
	263253268	2014-07-21	59.97	0.00	0.00	59.97	
	263253273	2014-07-22	30.75	0.00	0.00	30.75	▼

A SELECT statement that joins data from the Vendors and Invoices tables

```
SELECT vendor_name, invoice_number, invoice_date,  
       invoice_total  
FROM vendors INNER JOIN invoices  
     ON vendors.vendor_id = invoices.vendor_id  
WHERE invoice_total >= 500  
ORDER BY vendor_name, invoice_total DESC
```

The result set defined by the SELECT statement

vendor_name	invoice_number	invoice_date	invoice_total
Federal Express Corporation	963253230	2014-07-07	739.20
Ford Motor Credit Company	9982771	2014-07-24	503.20
Franchise Tax Board	RTR-72-3662-X	2014-05-25	1600.00
Fresno County Tax Collector	P02-88D77S7	2014-05-03	856.92
IBM	Q545443	2014-06-09	1083.58
Ingram	31359783	2014-06-03	1575.00
Ingram	31361833	2014-07-21	579.42
Malloy Lithographing Inc	0-2058	2014-05-28	37966.19

Terms to know

- Query
- Base table
- Result table (result set)
- Calculated value
- Join
- Inner join
- Outer join

A statement that adds a row to the Invoices table

```
INSERT INTO invoices
  (vendor_id, invoice_number, invoice_date,
   invoice_total, terms_id, invoice_due_date)
VALUES
  (12, '3289175', '2014-07-18', 165, 3, '2014-08-17')
```

A statement that changes the value of the `credit_total` column for a selected row

```
UPDATE invoices
SET credit_total = 35.89
WHERE invoice_number = '367447'
```

A statement that changes the values in the `invoice_due_date` column for all invoices with the specified `terms_id`

```
UPDATE invoices
SET invoice_due_date
    = DATE_ADD(invoice_due_date, INTERVAL 30 DAY)
WHERE terms_id = 4
```

A statement that deletes a selected invoice from the Invoices table

```
DELETE FROM invoices  
WHERE invoice_number = '4-342-8069'
```

A statement that deletes all paid invoices from the Invoices table

```
DELETE FROM invoices  
WHERE invoice_total - payment_total - credit_total = 0
```

A SELECT statement that's difficult to read

```
select invoice_number, invoice_date, invoice_total,  
payment_total, credit_total, invoice_total - payment_total -  
credit_total as balance_due from invoices where  
invoice_total - payment_total - credit_total > 0 order by  
invoice_date
```

A SELECT statement that's coded with a readable style

```
SELECT invoice_number, invoice_date, invoice_total,  
       payment_total, credit_total,  
       invoice_total - payment_total - credit_total  
       AS balance_due  
FROM invoices  
WHERE invoice_total - payment_total - credit_total > 0  
ORDER BY invoice_date
```

A SELECT statement with a block comment

```
/*  
Author: Joel Murach  
Date: 8/22/2014  
*/  
SELECT invoice_number, invoice_date, invoice_total,  
       invoice_total - payment_total - credit_total  
       AS balance_due  
FROM invoices
```

A SELECT statement with a single-line comment

```
-- The fourth column calculates the balance due  
SELECT invoice_number, invoice_date, invoice_total,  
       invoice_total - payment_total - credit_total  
       AS balance_due  
FROM invoices
```

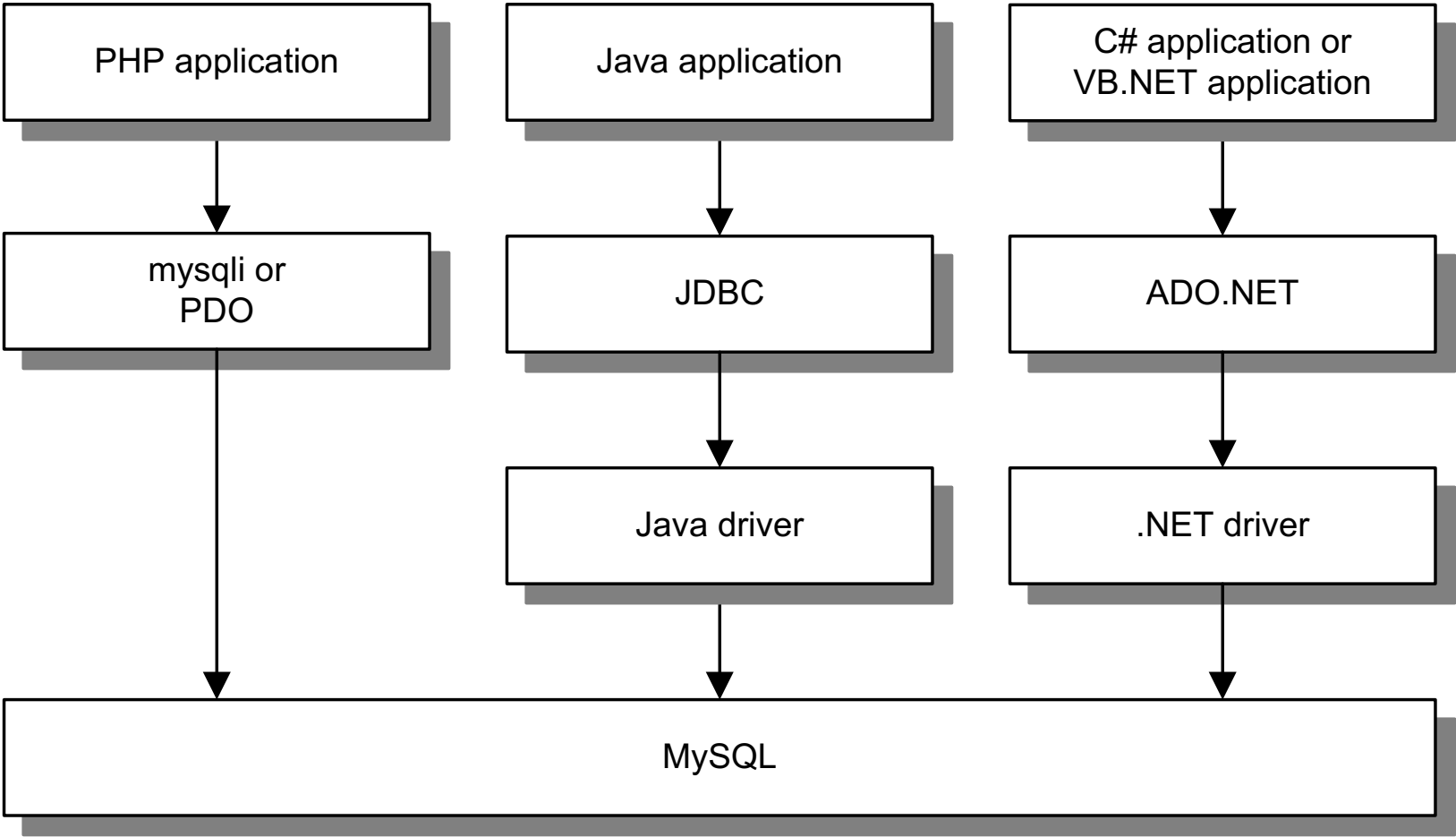
Coding recommendations

- Capitalize all keywords.
- Use lowercase for the other code.
- Separate the words in names with underscores.
- Start each clause on a new line.
- Break long clauses into multiple lines.
- Indent continued lines.
- Use comments only for code that is difficult to understand.
- Make sure that the comments are correct and up-to-date.

Note

- Line breaks, white space, indentation, and capitalization have no effect on the operation of a statement.

Common options for accessing MySQL data



Two commonly used MySQL drivers

- Connector/J
- Connector/Net

Terms to know

- Data access API
- mysqli API (for PHP)
- PDO API (for PHP)
- JDBC API (for Java)
- ADO.NET API (for .NET languages)
- Database driver

PHP code that gets data from MySQL

```
<?php
    $query =
        "SELECT vendor_name, invoice_number, invoice_total
        FROM vendors INNER JOIN invoices
            ON vendors.vendor_id = invoices.vendor_id
        WHERE invoice_total >= 500
        ORDER BY vendor_name, invoice_total DESC";

    $dsn = 'mysql:host=localhost;dbname=ap';
    $username = 'ap_tester';
    $password = 'sesame';

    try {
        $db = new PDO($dsn, $username, $password);
    } catch (PDOException $e) {
        $error_message = $e->getMessage();
        echo $error_message;
        exit();
    }
    $rows = $db->query($query);
?>
```

PHP code that gets data from MySQL (cont.)

```
<!DOCTYPE html>
<html>
  <head>
    <title>DB Test</title>
  </head>
  <body>
    <h1>Invoices with totals over 500:</h1>

    <?php foreach ($rows as $row) : ?>
    <p>
      Vendor: <?php echo
        $row['vendor_name']; ?><br/>
      Invoice No: <?php echo
        $row['invoice_number']; ?><br/>
      Total: $<?php echo
        number_format($row['invoice_total'], 2); ?>
    </p>
    <?php endforeach; ?>

  </body>
</html>
```

Java code that gets data from MySQL

```
package murach.ap;

import java.sql.*;
import java.text.NumberFormat;

public class DBTestApp {

    public static void main(String args[]) {
        String query
        = "SELECT vendor_name, invoice_number, invoice_total "
        + "FROM vendors INNER JOIN invoices "
        + "    ON vendors.vendor_id = invoices.vendor_id "
        + "WHERE invoice_total >= 500 "
        + "ORDER BY vendor_name, invoice_total DESC";

        String dbUrl = "jdbc:mysql://localhost:3306/ap";
        String username = "ap_tester";
        String password = "sesame";
```

Java code that gets data from MySQL (cont.)

```
// define common JDBC objects
try (Connection connection =
    DriverManager.getConnection(
        dbUrl, username, password);
    Statement statement =
        connection.createStatement();
    ResultSet rs = statement.executeQuery(query)) {

    // Display the results of a SELECT statement
    System.out.println(
        "Invoices with totals over 500:\n");
    while (rs.next()) {
        String vendorName =
            rs.getString("vendor_name");
        String invoiceNumber =
            rs.getString("invoice_number");
        double invoiceTotal =
            rs.getDouble("invoice_total");
```

Java code that gets data from MySQL (cont.)

```
        NumberFormat currency =
            NumberFormat.getCurrencyInstance();
        String invoiceTotalString =
            currency.format(invoiceTotal);

        System.out.println(
            "Vendor:      " + vendorName + "\n"
            + "Invoice No: " + invoiceNumber + "\n"
            + "Total:      " + invoiceTotalString
            + "\n");
    }
} catch (SQLException e) {
    System.out.println(e.getMessage());
}
}
}
```