Selections

Lecture 4

Robb T. Koether

Hampden-Sydney College

Mon, Jan 22, 2018

1. Datatypes
2. Constraints
3. Storage Engines
4. Selections
   - The `SELECT` Statement
   - Renaming Attributes
   - The `WHERE` Clause
   - The `ORDER BY` Clause

Date and Time Types

- Standard date and time types
  - `DATE` - A date in the format `'YYYY-MM-DD'`.
  - `TIME` - A time of day in the format `'HH:MM:SS'`.
  - `DATETIME` - A date and time in the format `'YYYY-MM-DD HH:MM:SS'`.
  - `TIMESTAMP` - A date and time in the format `'YYYY-MM-DD HH:MM:SS'`.
  - `YEAR(`n`)` - A year as either `'YYYY'` when $n = 4$ or `'YY'` when $n = 2$.

Date and Time Types

- The values of a `DATE` may range from `'1000-01-01'` to `'9999-12-31'`.
- The values of a `DATETIME` may range from `'1000-01-01 00:00:00'` to `'9999-12-31 23:59:59'`.
- The values of a `TIMESTAMP` may range from `'1970-01-01 00:00:01'` to `'2038-01-19 03:14:07'`.
- A `TIMESTAMP` attribute may be assigned the value `CURRENT_TIMESTAMP` or `NOW()`.

DATETIME Format

- There are several formats available for date input.

Format	Example
YYYY-MM-DD	'2013-01-05'
YYYY-M-DD	'2013-1-05'
YYYY-MM-D	'2013-01-5'
YYYY-M-D	'2013-1-5'
YY-MM-DD	'13-01-05'
YYYY/MM/DD	'2013/01/05'
YY/MM/DD	'13/01/05'
YYYY:MM:DD	'2013:01:05'
YYYYMMDD	'20130105'
YYMMDD	'130105'

`YEAR` Type

- The `YEAR` type represents only the year.
- There are two formats: `YYYY` and `YY`.
- If the year is given as a four-digit number `YYYY`, then is must be between 1901 and 2155.
- If the year is given as a two-digit number `YY`, then
  - 00 - 69 represent 2000 - 2069.
  - 70 - 99 represent 1970 - 1999.

YEAR Type

- The YEAR type represents only the year.
- There are two formats: YYYY and YY.
- If the year is given as a four-digit number YYYY, then is must be between 1901 and 2155.
- If the year is given as a two-digit number YY, then
  - 00 - 69 represent 2000 - 2069.
  - 70 - 99 represent 1970 - 1999.
- The YEAR(2) type has been removed as of version 5.7.

Default Values

Default Values

```
ssn CHAR(11) DEFAULT '000-00-0000';
```

- We may use the `DEFAULT` clause to specify a default value for an attribute when a value is not given.
- For example, in the `employees` relation, if the Social Security number is not given, then we may let `'000-00-0000'` be the default value.

Null Values

The `NOT NULL` Modifier

```
lname CHAR(20) NOT NULL;
```

- If we designate no default value and do not assign a value to an attribute in a tuple, then that attribute is given the value `NULL`.
- We may specify that a value may not be `NULL`.
- This prevents the value from being assigned `NULL` or being changed to `NULL`.
- For example, in the `employees` relation, we may specify that the last name be non-null.

The CHECK Clause

The CHECK Clause

```
dept TINYINT CHECK(dept >= 1 AND dept <= 5);
```

- Even though the value of the attribute is of the right type, we may want to check that it is in the right range.
- For example, in the `employees` relation, the department number must be between 1 and 5.
- This feature is not available in MySQL.

Primary Keys

The `PRIMARY KEY` Clause

```
PRIMARY KEY (ssn, dep_name)
```

- Every table should have a primary key.
- The primary key may be a single attribute or a set of attributes.
- A table's primary key must be unique for each tuple.
- For example, in the `dependents` relation, the Social Security number and name together must be unique. That would be a natural choice for the primary key.

Unique Values

The `UNIQUE` Modifier

```
dept_name VARCHAR(30) UNIQUE;
```

- We may use the `UNIQUE` modifier to specify that a given attribute will have a unique value for each tuple, even though it may not be a primary key.
- For example, in the `departments` relation, we may assume that each department has a unique name.

Storage Engines

- MySQL uses a storage engine to process queries.
- There are two main storage engines for MySQL.
  - MyISAM (My Indexed Sequential Access Method)
  - innoDB (Derived from Innobase, a Finnish company)
- The features available in MySQL and their performance depend on the storage engine.

Designating the Storage Engine

Designating the Storage Engine

```
CREATE TABLE table_name (attribute_list)
ENGINE = engine_name;
```

- When creating a table, after describing its attributes, specify the storage engine.
- If the storage engine is omitted, then the default is InnoDB.

The employees Table

The employees Table

```
CREATE TABLE employees (
    fname char(20) DEFAULT NULL,
    lname char(20) NOT NULL,
    ssn char(11) NOT NULL DEFAULT '000-00-0000',
    bdate date DEFAULT NULL,
    sex char(1) DEFAULT NULL,
    salary decimal(10,2) DEFAULT NULL,
    dept int(11) DEFAULT NULL,
    PRIMARY KEY (ssn)
) ENGINE=InnoDB DEFAULT CHARSET=latin1
```

Selections

The SELECT Statement

```
SELECT attribute_list
FROM table_name
WHERE condition
ORDER BY attribute_list;
```

- The SELECT command will select tuples from a relation.
- The attribute list can be *, which means all attributes.
- The WHERE and ORDER BY clauses are optional.

Selections

SELECT Example

```
SELECT * FROM employees;
+----------+---------+-----------+------------+------+-----------+------+
| fname    | lname   | ssn       | bdate      | sex  | salary    | dept |
+----------+---------+-----------+------------+------+-----------+------+
| Alice    | Smith   | 123456789 | 1968-05-22 | F    |  35000.00 |    2 |
| Barbara  | Brown   | 135792468 | 1985-10-12 | F    |  40000.00 |    3 |
| James    | Green   | 246813579 | 1974-02-15 | M    | 100000.00 |    1 |
| Jennifer | Wallace | 321549876 | 1985-12-02 | F    |  50000.00 |    2 |
| Frank    | Gilbert | 369147258 | 1966-08-21 | M    |  30000.00 |    3 |
| Joyce    | English | 456789012 | 1983-05-07 | F    |  25000.00 |    3 |
| Richard  | Johnson | 531978642 | 1955-03-17 | M    |  35000.00 |    3 |
| Ernest   | Roth    | 642189753 | 1986-06-12 | M    |  60000.00 |    2 |
| Susan    | Lane    | 654872109 | 1959-03-31 | F    |  70000.00 |    3 |
| John     | Kohler  | 789012345 | 1966-11-24 | M    |  40000.00 |    2 |
| Raymond  | Jones   | 963418527 | 1974-08-30 | M    |  80000.00 |    2 |
| Amy      | Ford    | 987105432 | 1968-01-21 | F    |  30000.00 |    3 |
+----------+---------+-----------+------------+------+-----------+------+
```

Selections

SELECT Example

```
SELECT fname, lname FROM employees;
+----------+----------+
| fname    | lname    |
+----------+----------+
| Alice    | Smith    |
| Barbara  | Brown    |
| James    | Green    |
| Jennifer | Wallace  |
| Frank    | Gilbert  |
| Joyce    | English  |
| Richard  | Johnson  |
| Ernest   | Roth     |
| Susan    | Lane     |
| John     | Kohler   |
| Raymond  | Jones    |
| Amy      | Ford     |
+----------+----------+
```

- We can specify only those attributes that we want to see.

Selections

SELECT Example

```
SELECT CONCAT(fname, ' ', lname) FROM employees;
+--------------------------+
| CONCAT(fname, ' ', lname) |
+--------------------------+
| Alice Smith              |
| Barbara Brown            |
| James Green              |
| Jennifer Wallace         |
| Frank Gilbert            |
| Joyce English            |
| Richard Johnson          |
| Ernest Roth              |
| Susan Lane               |
| John Kohler              |
| Raymond Jones            |
| Amy Ford                 |
+--------------------------+
```

- In the attribute list, we may write expressions as well as simple attributes.

Renaming Attributes

SELECT Example

```
SELECT CONCAT(fname, ' ', lname) AS emp_name FROM employees;
+------------------+
| emp_name         |
+------------------+
| Alice Smith      |
| Barbara Brown    |
| James Green      |
| Jennifer Wallace |
| Frank Gilbert    |
| Joyce English    |
| Richard Johnson  |
| Ernest Roth      |
| Susan Lane       |
| John Kohler      |
| Raymond Jones    |
| Amy Ford         |
+------------------+
```

- Using the `AS` clause, we can rename the "attribute."

The WHERE Clause

WHERE Example

```
SELECT fname, lname, salary FROM employees
WHERE salary > 40000;
+----------+---------+-----------+
| fname    | lname   | salary    |
+----------+---------+-----------+
| James    | Green   | 100000.00 |
| Jennifer | Wallace |  50000.00 |
| Ernest   | Roth    |  60000.00 |
| Susan    | Lane    |  70000.00 |
| Raymond  | Jones   |  80000.00 |
+----------+---------+-----------+
```

- In the WHERE clause, we write a boolean expression that will filter out some of the tuples.
- Use AND, OR, and NOT for compound expressions.

The WHERE Clause

WHERE Example

```
SELECT fname, lname, sex, bdate, salary
FROM employees
WHERE (bdate >= '1982-01-01' AND sex = 'M')
OR (bdate >= '1976-01-01' AND sex = 'F');
+----------+---------+------+------------+----------+
| fname    | lname   | sex  | bdate      | salary   |
+----------+---------+------+------------+----------+
| Barbara  | Brown   | F    | 1985-10-12 | 40000.00 |
| Jennifer | Wallace | F    | 1985-12-02 | 50000.00 |
| Joyce    | English | F    | 1983-05-07 | 25000.00 |
| Ernest   | Roth    | M    | 1986-06-12 | 60000.00 |
+----------+---------+------+------------+----------+
```

The WHERE Clause

WHERE Example

```
SELECT fname, lname, sex, bdate, salary
FROM employees
WHERE NOT((bdate >= '1982-01-01' AND sex = 'M')
OR (bdate >= '1976-01-01' AND sex = 'F'));
```

fname	lname	sex	bdate	salary
Alice	Smith	F	1968-05-22	35000.00
James	Green	M	1974-02-15	100000.00
Frank	Gilbert	M	1966-08-21	30000.00
Richard	Johnson	M	1955-03-17	35000.00
Susan	Lane	F	1959-03-31	70000.00
John	Kohler	M	1966-11-24	40000.00
Raymond	Jones	M	1974-08-30	80000.00
Amy	Ford	F	1968-01-21	30000.00

The WHERE Clause

WHERE Example

```
SELECT fname, lname, sex, bdate, salary
FROM employees
WHERE (bdate < '1982-01-01' OR sex = 'F')
AND (bdate < '1976-01-01' OR sex = 'M');
```

fname	lname	sex	bdate	salary
Alice	Smith	F	1968-05-22	35000.00
James	Green	M	1974-02-15	100000.00
Frank	Gilbert	M	1966-08-21	30000.00
Richard	Johnson	M	1955-03-17	35000.00
Susan	Lane	F	1959-03-31	70000.00
John	Kohler	M	1966-11-24	40000.00
Raymond	Jones	M	1974-08-30	80000.00
Amy	Ford	F	1968-01-21	30000.00

The ORDER BY Clause

ORDER BY Example

```
SELECT fname, lname, salary FROM employees ORDER BY salary;
+----------+---------+-----------+
| fname    | lname   | salary    |
+----------+---------+-----------+
| Joyce    | English |  25000.00 |
| Amy      | Ford    |  30000.00 |
| Frank    | Gilbert |  30000.00 |
| Richard  | Johnson |  35000.00 |
| Alice    | Smith   |  35000.00 |
| Barbara  | Brown   |  40000.00 |
| John     | Kohler  |  40000.00 |
| Jennifer | Wallace |  50000.00 |
| Ernest   | Roth    |  60000.00 |
| Susan    | Lane    |  70000.00 |
| Raymond  | Jones   |  80000.00 |
| James    | Green   | 100000.00 |
+----------+---------+-----------+
```

- We can use the ORDER BY clause to sort the tuples by specified fields.

The ORDER BY Clause

ORDER BY Example

```
SELECT fname, lname, salary FROM employees
WHERE sex = 'F' ORDER BY salary;
+-----------+----------+-----------+
| fname     | lname    | salary    |
+-----------+----------+-----------+
| Joyce     | English  | 25000.00  |
| Amy       | Ford     | 30000.00  |
| Alice     | Smith    | 35000.00  |
| Barbara   | Brown    | 40000.00  |
| Jennifer  | Wallace  | 50000.00  |
| Susan     | Lane     | 70000.00  |
+-----------+----------+-----------+
```

- The ORDER BY clause comes after the WHERE clause.

The ORDER BY Clause

ORDER BY Example

```
SELECT fname, lname, sex FROM employees ORDER BY sex, lname;
+----------+---------+------+
| fname    | lname   | sex  |
+----------+---------+------+
| Barbara  | Brown   | F    |
| Joyce    | English | F    |
| Amy      | Ford    | F    |
| Susan    | Lane    | F    |
| Alice    | Smith   | F    |
| Jennifer | Wallace | F    |
| Frank    | Gilbert | M    |
| James    | Green   | M    |
| Richard  | Johnson | M    |
| Raymond  | Jones   | M    |
| John     | Kohler  | M    |
| Ernest   | Roth    | M    |
+----------+---------+------+
```

- If we specify more than one attribute, then they are treated as major and minor sort fields.

The ORDER BY Clause

ORDER BY Example

```
SELECT fname, lname, sex, salary FROM employees
ORDER BY sex, salary DESC;
+----------+---------+------+-----------+
| fname    | lname   | sex  | salary    |
+----------+---------+------+-----------+
| Susan    | Lane    | F    |  70000.00 |
| Jennifer | Wallace | F    |  50000.00 |
| Barbara  | Brown   | F    |  40000.00 |
| Alice    | Smith   | F    |  35000.00 |
| Amy      | Ford    | F    |  30000.00 |
| Joyce    | English | F    |  25000.00 |
| James    | Green   | M    | 100000.00 |
| Raymond  | Jones   | M    |  80000.00 |
| Ernest   | Roth    | M    |  60000.00 |
| John     | Kohler  | M    |  40000.00 |
| Richard  | Johnson | M    |  35000.00 |
| Frank    | Gilbert | M    |  30000.00 |
+----------+---------+------+-----------+
```

- We can specify that any sort field be sorted in *descending* order.