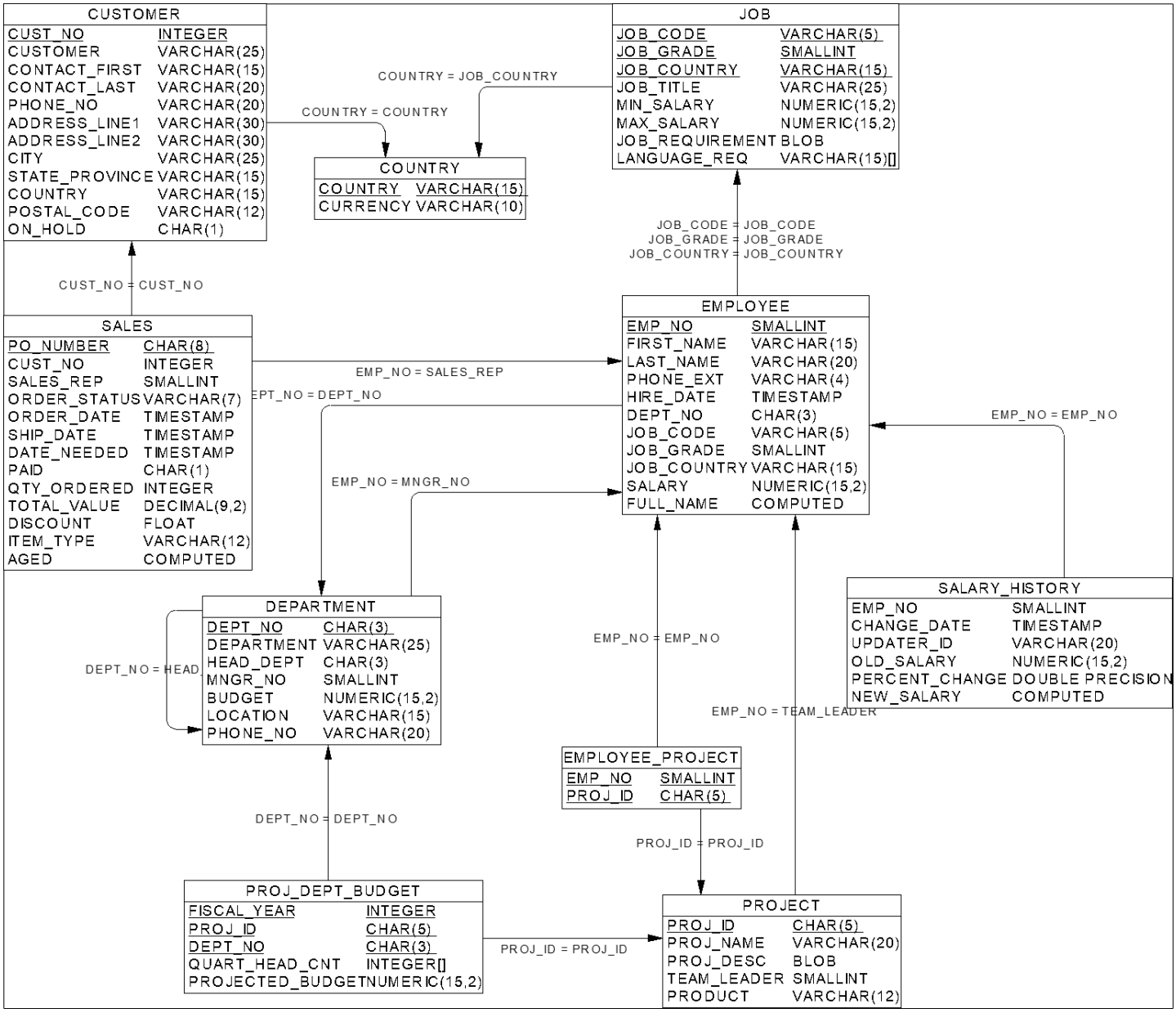
1. Demo database for the course: EMPLOYEE
   (a sample database included with the distribution of Firebird Server, adapted to MySQL).

Database schema



Sample tables and data from database employee

EMPLOYEE

EMP_NO	FIRST_NAME	LAST_NAME	PHONE_EXT	HIRE_DATE	DEPT_NO	JOB_CODE	JOB_GRADE	JOB_COUNTRY	SALARY	FULL_NAME
2	Robert	Nelson	250	1988-12-28	600	VP	2	USA	105 900,00	Nelson, Robert
4	Bruce	Young	233	1988-12-28	621	Eng	2	USA	97 500,00	Young, Bruce
5	Kim	Lambert	22	1989-02-06	130	Eng	2	USA	102 750,00	Lambert, Kim
8	Leslie	Johnson	410	1989-04-05	180	Mktg	3	USA	64 635,00	Johnson, Leslie
9	Phil	Forest	229	1989-04-17	622	Mngr	3	USA	75 060,00	Forest, Phil
11	K. J.	Weston	34	1990-01-17	130	SRep	4	USA	86 292,94	Weston, K. J.
12	Terri	Lee	256	1990-05-01	000	Admin	4	USA	54 793,00	Lee, Terri
14	Stewart	Hall	227	1990-06-04	900	Finan	3	USA	69 482,63	Hall, Stewart
15	Katherine	Young	231	1990-06-14	623	Mngr	3	USA	67 241,25	Young, Katherine
20	Chris	Papadopoulos	887	1990-01-01	671	Mngr	3	USA	89 655,00	Papadopoulos, Chris
24	Pete	Fisher	888	1990-09-12	671	Eng	3	USA	81 810,19	Fisher, Pete
28	Ann	Bennet	5	1991-02-01	120	Admin	5	England	22 935,00	Bennet, Ann
29	Roger	De Souza	288	1991-02-18	623	Eng	3	USA	69 482,63	De Souza, Roger
34	Janet	Baldwin	2	1991-03-21	110	Sales	3	USA	61 637,81	Baldwin, Janet
36	Roger	Reeves	6	1991-04-25	120	Sales	3	England	33 620,63	Reeves, Roger
37	Willie	Stansbury	7	1991-04-25	120	Eng	4	England	39 224,06	Stansbury, Willie
44	Leslie	Phong	216	1991-06-03	623	Eng	4	USA	56 034,38	Phong, Leslie
45	Ashok	Ramanathan	209	1991-08-01	621	Eng	3	USA	80 689,50	Ramanathan, Ashok
46	Walter	Steadman	210	1991-08-09	900	CFO	1	USA	116 100,00	Steadman, Walter
52	Carol	Nordstrom	420	1991-10-02	180	PRel	4	USA	42 742,50	Nordstrom, Carol
61	Luke	Leung	3	1992-02-18	110	SRep	4	USA	68 805,00	Leung, Luke

DEPARTMENT

DEPT_NO	DEPARTMENT	HEAD_DEPT	MNGR_NO	BUDGET	LOCATION	PHONE_NO
000	Corporate Headquarters	Null	105	1 000 000,00	Monterey	(408) 555-1234
100	Sales and Marketing	000	85	2 000 000,00	San Francisco	(415) 555-1234
600	Engineering	000	2	1 100 000,00	Monterey	(408) 555-1234
900	Finance	000	46	400 000,00	Monterey	(408) 555-1234
180	Marketing	100	Null	1 500 000,00	San Francisco	(415) 555-1234
620	Software Products Div.	600	Null	1 200 000,00	Monterey	(408) 555-1234
621	Software Development	620	Null	400 000,00	Monterey	(408) 555-1234
622	Quality Assurance	620	9	300 000,00	Monterey	(408) 555-1234
623	Customer Support	620	15	650 000,00	Monterey	(408) 555-1234
670	Consumer Electronics Div.	600	107	1 150 000,00	Burlington, VT	(802) 555-1234
671	Research and Development	670	20	460 000,00	Burlington, VT	(802) 555-1234
672	Customer Services	670	94	850 000,00	Burlington, VT	(802) 555-1234
130	Field Office: East Coast	100	11	500 000,00	Boston	(617) 555-1234
140	Field Office: Canada	100	72	500 000,00	Toronto	(416) 677-1000
110	Pacific Rim Headquarters	100	34	600 000,00	Kuaui	(808) 555-1234
115	Field Office: Japan	110	118	500 000,00	Tokyo	3 5350 0901
116	Field Office: Singapore	110	Null	300 000,00	Singapore	3 55 1234
120	European Headquarters	100	36	700 000,00	London	71 235-4400
121	Field Office: Switzerland	120	141	500 000,00	Zurich	1 211 7767
123	Field Office: France	120	134	400 000,00	Cannes	58 68 11 12
125	Field Office: Italy	120	121	400 000,00	Milan	2 430 39 39

PROJECT

PROJ_ID	PROJ_NAME	PROJ_DESC	TEAM_LEADER	PRODUCT
VBASE	Video Database	Design a video data base management system for	45	software
DGPII	DigiPizza	Develop second generation digital pizza maker	24	other
GUIDE	AutoMap	Develop a prototype for the automobile version of	20	hardware
MAPDB	MapBrowser port	Port the map browsing database software to run	4	software
HWRII	Translator upgrade	Integrate the hand-writing recognition module into the	Null	software
MKTPR	Marketing project 3	Expand marketing and sales in the Pacific Rim.	85	N/A

2. SQL SELECT Statement

- The SELECT statement is used to extract data from a database (from a table, view or other database objects).
- The result is stored in a result table, called the result-set.
- The result set table has columns, as specified in the SELECT statement, and rows, which satisfies the imposed conditions.
- The set of rows return by the SELECT statement may contain duplicates.
- The syntax is rather complicated. There can be several levels of nesting.
- The user to execute a SELECT statement must be granted appropriate privileges to select data.
- SQL keywords, names of tables, columns, etc., are not case-sensitive.
- A semicolon at the end of each SQL statement is not required.

Simplified syntax of the SELECT statement

SELECT ..... *columns names, expressions, functions (separated by a comma)*
FROM ..... *tables or views names, joining clauses*
WHERE ..... *the condition used to filter records*
GROUP BY ..... *columns names, according to which the result set will be grouped by*
HAVING ..... *the condition used to filter groups*
ORDER BY ..... *columns (or expressions) the result set is sorted by*

*a) The below statement selects all rows and all columns from the table employee (use an asterisk * to choose all columns from a table)*

```
SELECT  *
FROM employee
```

EMP_NO	FIRST_NAME	LAST_NAME	PHONE_EXT	HIRE_DATE	DEPT_NO	JOB_CODE	JOB_GRADE	JOB_COUNTRY	SALARY	FULL_NAME
2	Robert	Nelson	250	1988-12-28	600	VP	2	USA	105 900,00	Nelson, Robert
4	Bruce	Young	233	1988-12-28	621	Eng	2	USA	97 500,00	Young, Bruce
5	Kim	Lambert	22	1989-02-06	130	Eng	2	USA	102 750,00	Lambert, Kim
8	Leslie	Johnson	410	1989-04-05	180	Mktg	3	USA	64 635,00	Johnson, Leslie
9	Phil	Forest	229	1989-04-17	622	Mngr	3	USA	75 060,00	Forest, Phil
11	K. J.	Weston	34	1990-01-17	130	SRep	4	USA	86 292,94	Weston, K. J.

b) Choosing only selected columns

```
SELECT full_name, salary, hire_date
FROM employee
```

FULL_NAME	SALARY	HIRE_DATE
Nelson, Robert	105 900,00	1988-12-28
Young, Bruce	97 500,00	1988-12-28
Lambert, Kim	102 750,00	1989-02-06
Johnson, Leslie	64 635,00	1989-04-05
Forest, Phil	75 060,00	1989-04-17
Weston, K. J.	86 292,94	1990-01-17
Lee, Terri	54 793,00	1990-05-01

```
SELECT full_name AS employee_name, hire_date AS date, salary AS year_salary
FROM employee
```

EMPLOYEE_NAME	HIRE_DATE	YEAR_SALARY
Nelson, Robert	1988-12-28	105 900,00
Young, Bruce	1988-12-28	97 500,00
Lambert, Kim	1989-02-06	102 750,00
Johnson, Leslie	1989-04-05	64 635,00
Forest, Phil	1989-04-17	75 060,00
Weston, K. J.	1990-01-17	86 292,94
Lee, Terri	1990-05-01	54 793,00
Hall, Stewart	1990-06-04	69 482,63
Young, Katherine	1990-06-14	67 241,25
Papadopoulos, Chris	1990-01-01	89 655,00

To each SQL expression and any column in the select clause (and to a table as well), one can give an alias (used to temporarily rename a table or a column heading).

c) Ordering the result set: clause ORDER BY

Ascending sort, default, option ASC can be skipped

```
SELECT full_name, salary, hire_date
FROM employee
ORDER BY full_name ASC
```

FULL_NAME	SALARY	HIRE_DATE
Baldwin, Janet	61 637,81	1991-03-21
Bender, Oliver H.	212 850,00	1992-10-08
Bennet, Ann	22 935,00	1991-02-01
Bishop, Dana	62 550,00	1992-06-01
Brown, Kelly	27 000,00	1993-02-04
Burbank, Jennifer M.	53 167,50	1992-04-15
Cook, Kevin	111 262,50	1993-02-01
De Souza, Roger	69 482,63	1991-02-18

For a descending sort, use option DESC

```
SELECT full_name, salary, hire_date
FROM employee
ORDER BY salary DESC
```

FULL_NAME	SALARY	HIRE_DATE
Ferrari, Roberto	99 000 000,00	1993-07-12
Yamamoto, Takashi	7 480 000,00	1993-07-01
Ichida, Yuki	6 000 000,00	1993-02-04
Glon, Jacques	390 500,00	1993-08-23
Bender, Oliver H.	212 850,00	1992-10-08
Steadman, Walter	116 100,00	1991-08-09
MacDonald, Mary S.	111 262,50	1992-06-01
Cook, Kevin	111 262,50	1993-02-01

Ascending sort, by two columns

```
SELECT full_name, salary, hire_date
FROM employee
ORDER BY last_name, salary
```

d) SQL expressions

- An SQL expression can be composed from columns names, operators, constants and functions.
- Binary operators: arithmetical +, -, *,/.
- SQL uses standard order of operators, brackets (and) can be used if necessary.
- To each SQL expression and any column in the select clause (and to a table as well), one can give an *alias* (used to temporarily rename a table or a column heading). If an alias consists of two or more words, use quotation marks, e.g., "salary per month".

```
SELECT full_name, salary AS actual_salary, salary*1.2 AS rise, (salary+200)/100 AS tax
FROM employee
```

FULL_NAME	ACTUAL_SALARY	RISE	TAX
Nelson, Robert	105 900,00	127 080,000	1 061,00
Young, Bruce	97 500,00	117 000,000	977,00
Lambert, Kim	102 750,00	123 300,000	1 029,50
Johnson, Leslie	64 635,00	77 562,000	648,35
Forest, Phil	75 060,00	90 072,000	752,60
Weston, K. J.	86 292,94	103 551,528	864,92
Lee, Terri	54 793,00	65 751,600	549,93
Hall, Stewart	69 482,63	83 379,156	696,82

In this statement, three aliases (actual_salary, rise, tax) are created (for the column salary, and next two expressions, respectively).

```
SELECT CONCAT(last_name,'   ',phone_ext) AS contact_data        -- built-in function CONCAT()
FROM employee
```

e) Use DISTINCT to eliminate duplicates

```
SELECT DISTINCT job_country
FROM employee
```

JOB_COUNTRY
Canada
England
France
Italy
Japan
Switzerland
USA

3. SQL SELECT Statement with the clause WHERE

- The WHERE clause is used to filter records. It is used to extract only those records that fulfil a specified criterion – the condition which can be made of:
  - names of columns, functions, constants,
  - operators of comparison =, <, >, <>, <=, >=, !=
  - SQL operators such as LIKE, BETWEEN, IN
  - logical operators AND &&, OR ||, NOT !, XOR
- the condition may return one of the values: *true, false, unknown (NULL)* (if the condition returns unknown, it is often caused by the occurrence of *NULL*, which means an empty value),
- the SELECT statement will return the records, for which the condition in WHERE evaluated to *true*,
- SQL requires single quotes around text values and data/time values; no quotes around numeric fields,
- in the WHERE clause we cannot use aggregating functions,
- the conditions in WHERE can be constructed using nested queries, i.e., *subselects*.

a) WHERE with a compound condition using logical operators

```
SELECT full_name, hire_date, job_country
FROM employee
WHERE job_country='Japan' OR job_country='Italy'
```

FULL_NAME	HIRE_DATE	JOB_COUNTRY
Ichida, Yuki	1993-02-04	Japan
Yamamoto, Takashi	1993-07-01	Japan
Ferrari, Roberto	1993-07-12	Italy

```
SELECT full_name, hire_date, job_country
FROM employee
WHERE hire_date>'01.07.1993' AND
        job_country<>'USA'
```

FULL_NAME	HIRE_DATE	JOB_COUNTRY
Ferrari, Roberto	1993-07-12	Italy
Glon, Jacques	1993-08-23	France
Osborne, Pierre	1994-01-03	Switzerland

b) SQL operators

- Operator IN - checks, whether the value belongs to the specified set
  value IN (value1, value2, …)

```
SELECT *
FROM employee
WHERE job_country IN ('USA' , 'Italy')
```

- Operator BETWEEN – checks, whether the value is contained in a specified closed interval
  value BETWEEN value1 AND value2

```
SELECT *
FROM employee
WHERE salary BETWEEN 50000 AND 100000
```

- To check, if the value is an empty value NULL or not, use operator IS NULL or IS NOT NULL
  value IS NULL

```
SELECT *
FROM employee
WHERE phone_ext IS NULL
```

- The LIKE operator is used to search for a specified pattern in a column.
  text expression LIKE ‘pattern’
  There are two wildcard characters which are used to create a pattern in the LIKE operator:
  % a substitute for zero or more characters
  _ a substitute for a single character

The next query searches for all employees whose name starts with *B*:

```
SELECT *
FROM employee
WHERE last_name LIKE 'B%'
```

To list all employees whose last name ends on *la*

```
SELECT *
FROM employee
WHERE last_name LIKE '%la'
```

The next query searches for all employees whose name starts with *B*, next two characters are arbitrary, then it must contain letter *x*, then any characters:

```
SELECT *
FROM employee
WHERE last_name LIKE 'B__x%'
```

If the text we are looking for contains _ or %, we can use ESCAPE
For example, to search for a sign _ in a column phone_ext, we can use LIKE as follows:

```
SELECT *
FROM employee
WHERE phone_ext LIKE '%^_%' ESCAPE '^'
```

SQL comments:

Single-line:
-- comment

Multi-line:
/* ……
……
……. ……*/