SQL

Key Points

- SQL is the standard language for interacting with RDBMS
  - supported by databases from many different vendors (though there are incompatibilities due to incomplete implementations and non-standard extensions)
- bags vs sets
  - in the relational model, relations are sets
    - no duplicate tuples
    - order of tuples doesn't matter
  - in practice, tables are implemented as bags
    - order of rows doesn't matter
    - *duplicates are not automatically removed from query results*
- basic queries
  - SELECT … FROM … WHERE …
    - JOIN, NATURAL JOIN
  - DISTINCT, ORDER BY
  - NULL values
- notation used for syntax in the MySQL documentation

Basic Queries

```
SELECT a1,a2,...,an
FROM R
WHERE C
```

- (only) table and column names are case-sensitive
- resulting table has
  - the rows from R for which C is true
  - the columns a1,a2,…,an

Basic Queries

```
SELECT a1,a2,...,an
FROM R
WHERE C
ORDER BY b1,b2,...,bm
```

- rows are sorted according to the specified columns just before the SELECT is applied (so b1,b2,…,bm can involve columns other than a1,a2,…,an)
- default is ascending order; use DESC for descending order

```
SELECT DISTINCT a1,a2,...,an
FROM R
WHERE C
```

- as the last step, remove duplicate rows (duplicates identified based only on columns a1,a2,...an)

Basic Queries

SELECT
- can contain * (all columns), arithmetic expressions
- can rename individual columns of the result using `AS`

FROM
- can rename the whole relation using `AS`
- can contain `R,S` (cross product), `R JOIN S ON C`, `R NATURAL JOIN S`
  - `JOIN` keeps all columns of R and S – equivalent to `FROM R,S WHERE C`
  - `NATURAL JOIN` keeps only one copy of each join column
- qualify column name with relation name or alias to disambiguate same-named columns

WHERE
- can contain arithmetic expressions, <, <=, >, >=, =, <>, `AND`, `OR`, `NOT`, `IS NULL`, `IS NOT NULL`, `LIKE`, `NOT LIKE`
  - wildcards % (0 or more), _ (single character) allowed in `LIKE`/`NOT LIKE` patterns – used for pattern-matching within column values

Basic Queries

Notes on data types.
- ' ' (single quotes) denote string values
- DATE – 'yyyy-mm-dd'
- TIME – 'hh:mm:ss' or 'hh:mm:ss+h:mm' or 'hh:mm:ss-h:mm' (relative to GMT)
- TIMESTAMP – 'yyyy-mm-dd hh:mm:ss'

NULL values.
- any operation applied to NULL yields NULL
- any comparison with NULL yields UNKNOWN

Three-valued logic.
- works out as if TRUE = 1, FALSE = 0, UNKNOWN = ½ and AND = min, OR = max, NOT = 1-value

Syntax Notation

- blue (and not italics) – type literally that
- *italics* – replace with the appropriate specifics of that kind of thing
- | - list of alternatives, choose one
- [ ] - optional
- { } - provides grouping when there are alternatives for a required thing (don't type the {} themselves)

```
SELECT
    [ALL | DISTINCT | DISTINCTROW ]
    [HIGH_PRIORITY]
    [STRAIGHT_JOIN]
    [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT]
    [SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS]
    select_expr [, select_expr] ...
    [into_option]
    [FROM table_references
      [PARTITION partition_list]]
    [WHERE where_condition]
    [GROUP BY {col_name | expr | position}, ... [WITH ROLLUP]]
    [HAVING where_condition]
    [WINDOW window_name AS (window_spec)
        [, window_name AS (window_spec)] ...]
    [ORDER BY {col_name | expr | position}
      [ASC | DESC], ... [WITH ROLLUP]]
    [LIMIT {[offset,] row_count | row_count OFFSET offset}]
    [into_option]
    [FOR {UPDATE | SHARE}
        [OF tbl_name [, tbl_name] ...]
        [NOWAIT | SKIP LOCKED]
      | LOCK IN SHARE MODE]
    [into_option]

into_option: {
    INTO OUTFILE 'file_name'
        [CHARACTER SET charset_name]
        export_options
  | INTO DUMPFILE 'file_name'
  | INTO var_name [, var_name] ...
}
```


Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

```
SELECT DISTINCT Dependent_name
FROM DEPENDENT
WHERE Sex='F'

SELECT *
FROM DEPT_LOCATIONS JOIN PROJECT ON Dnumber=Dnum

SELECT *
FROM WORKS_ON JOIN PROJECT ON Pno=Pnmber
WHERE Plocation='Houston' AND Hours >= 10

SELECT Fname,Lname
FROM EMPLOYEE JOIN DEPARTMENT ON Super_ssn=Mgr_ssn
WHERE Dno=Dnumber

SELECT Fname,Lname,Dependent_name,Relationship
FROM EMPLOYEE NATURAL JOIN DEPENDENT

SELECT *
FROM DEPARTMENT NATURAL JOIN DEPT_LOCATIONS
```


Other Joins

```
  table_reference {[INNER | CROSS] JOIN | STRAIGHT_JOIN} table_factor [join_specification]
| table_reference {LEFT|RIGHT} [OUTER] JOIN table_reference join_specification
| table_reference NATURAL [INNER | {LEFT|RIGHT} [OUTER]] JOIN table_factor
```

- `R LEFT JOIN S`
  - if there is no matching row for R in S, a row with all `NULL`s is used as the match
  - every row of R is represented in the result
  - `R RIGHT JOIN S` is equivalent to `S LEFT JOIN R`
    - `LEFT JOIN` is recommended for portability
- (full) `OUTER JOIN`
  - if a row in either table doesn't have a match in the other, a row with all `NULL`s is used as the match
  - MySQL does not (directly) support a full outer join
- `CROSS JOIN`, `INNER JOIN`
  - syntactically equivalent to `JOIN` in MySQL (but not in standard SQL)
- `R STRAIGHT_JOIN S`
  - R is always read first – gives directions to the optimizer when the query is processed

Strategies for Writing Queries

- start with FROM – build a table that contains all of the rows and columns that you need
- add WHERE to pick the desired rows
  - prefer JOIN condition to WHERE condition when specifying how rows of one table combine with rows of another
- add SELECT to pick the desired columns