Relational Database

- A relational database is a collection of relations or two-dimensional tables.

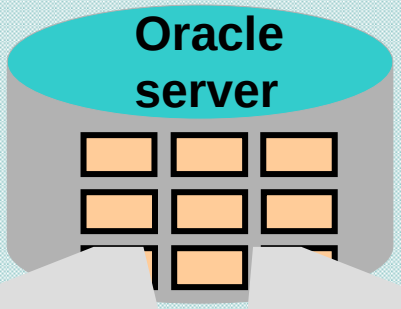


Table name: `EMPLOYEES`

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL
100	Steven	King	SKING
101	Neena	Kochhar	NKOCHHAR
102	Lex	De Haan	LDEHAAN

...

Table name: `DEPARTMENTS`

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID
10	Administration	200
20	Marketing	201
50	Shipping	124

...

Relational Database Terminology

EMPLOYEE_ID	FIRST_NAME	LAST_NAME	SALARY	COMMISSION_PCT	DEPARTMENT_ID
100	Steven	King	24000	(null)	90
101	Neena	Kochhar	17000	(null)	90
102	Lex	De Haan	17000	(null)	90
103	Alexander	Hunold	9000	(null)	60
104	Bruce	Ernst	6000	(null)	60
107	Diana	Lorentz	4200	(null)	60
124	Kevin	Mourgos	5800	(null)	50
141	Trenna	Rajs	3500	(null)	50
142	Curtis	Davies	3100	(null)	50
143	Randall	Matos	2600	(null)	50
144	Peter	Vargas	2500	(null)	50
149	Eleni	Zlotkey	10500	0.2	80
174	Ellen	Abel	11000	0.3	80
176	Jonathon	Taylor	8600	0.2	80
178	Kimberely	Grant	7000	0.15	(null)
200	Jennifer	Whalen	4400	(null)	10
201	Michael	Hartstein	13000	(null)	20
202	Pat	Fay	6000	(null)	20
205	Shelley	Higgins	12000	(null)	110
206	William	Gietz	8300	(null)	110

1 2 3 4 5 6

SQL

SQL Statements

- SELECT
- INSERT
- UPDATE
- DELETE

Data manipulation language (DML)

- CREATE
- ALTER
- DROP
- RENAME
- TRUNCATE
- COMMENT

Data definition language (DDL)

- GRANT
- REVOKE

Data control language (DCL)

- COMMIT
- ROLLBACK
- SAVEPOINT

Transaction control

Structured Query Language

- ✓ SQL is a tool for organizing, managing, and retrieving data stored by a database.
- ✓ SQL is the set of statements with which all programs and users access data in a database.
- ✓ SQL has been accepted as the standard RDBMS language all over the world
- ✓ SQl provides an interface between relational database such as Oracle, Sybase, Ms SQL etc and user.
- ✓ All SQL statements are instructions to the database
- ✓ SQL is a non procedural language, means access method of data is not required to be specified.
- ✓ All SQL statements use the optimizer, a part of Oracle that determines the most efficient means of accessing the specified data
- ✓ SQL processes sets of data as groups rather than as individual

SQL Roles and Benefits

- ✓ SQL is an interactive query language.
- ✓ SQL is Database programming language.
- ✓ SQL is Database administration language.
- ✓ SQL is Client/server language.
- ✓ SQL is Internet data access language.
- ✓ SQL is Distributed database language.
- ✓ SQL is Database gateway language.
- ✓ Vendor Independence.
- ✓ Portability across computer system.
- ✓ Multiple views of data.
- ✓ High-Level English-Like structure.

Database Objects

Object	Description
Table	Basic unit of storage; composed of rows and columns
View	Logically represents subsets of data from one or more tables
Sequence	Numeric value generator
Index	Improves the performance of some queries
Synonym	Gives alternative names to objects

Naming Rules

- Name length - Names for databases, tables, columns, and indexes can be up to 64 characters long. Alias names can be up to 256 characters long.
- Name qualifiers - Depending on context, a name may need to be qualified to make it clear what the name refers to. To refer to a database, just specify its name.

MySQL Data types

- String Types
- Numeric Types
- Date and Time Types

String data types

- CHAR(size) - Holds a fixed length string (can contain letters, numbers, and special characters). The fixed size is specified in parenthesis. Can store up to 255 characters
-
- VARCHAR(size) - Holds a variable length string (can contain letters, numbers, and special characters). The maximum size is specified in parenthesis. Can store up to 255 characters. **Note:** If you put a greater value than 255 it will be converted to a TEXT type
- TINYTEXT - Holds a string with a maximum length of 255 characters
- TEXT - Holds a string with a maximum length of 65,535 characters
- BLOB - For BLOBs (Binary Large OBjects). Holds up to 65,535 bytes of data
- MEDIUMTEXT - Holds a string with a maximum length of 16,777,215 characters
- MEDIUMBLOB - For BLOBs (Binary Large OBjects). Holds up to 16,777,215 bytes of data

String data types

- LONGTEXT - Holds a string with a maximum length of 4,294,967,295 characters
- LONGBLOB - For BLOBs (Binary Large OBjects). Holds up to 4,294,967,295 bytes of data
- ENUM(x,y,z,etc.) - Let you enter a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted.
- **Note:** The values are sorted in the order you enter them.
- SET - You enter the possible values in this format: ENUM('X','Y','Z') .Similar to ENUM except that SET may contain up to 64 list items and can store more than one choice

Number data types

Data type	Description
TINYINT(size)	-128 to 127 normal. 0 to 255 UNSIGNED*. The maximum number of digits may be specified in parenthesis
SMALLINT(size)	-32768 to 32767 normal. 0 to 65535 UNSIGNED*. The maximum number of digits may be specified in parenthesis
MEDIUMINT(size)	-8388608 to 8388607 normal. 0 to 16777215 UNSIGNED*. The maximum number of digits may be specified in parenthesis

Number data types

INT(size)	-2147483648 to 2147483647 normal. 0 to 4294967295 UNSIGNED*. The maximum number of digits may be specified in parenthesis
BIGINT(size)	-9223372036854775808 to 9223372036854775807 normal. 0 to 18446744073709551615 UNSIGNED*. The maximum number of digits may be specified in parenthesis
FLOAT(size,d)	A small number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DOUBLE(size,d)	A large number with a floating decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter
DECIMAL(size,d)	A DOUBLE stored as a string , allowing for a fixed decimal point. The maximum number of digits may be specified in the size parameter. The maximum number of digits to the right of the decimal point is specified in the d parameter

*The integer types have an extra option called UNSIGNED. Normally, the integer goes from an negative to positive value. Adding the UNSIGNED attribute will move that range up so it starts at zero instead of a negative number.

Date data types

Data type	Description
DATE()	A date. Format: YYYY-MM-DD **Note:** The supported range is from '1000-01-01' to '9999-12-31'
DATETIME()	*A date and time combination. Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1000-01-01 00:00:00' to '9999-12-31 23:59:59'
TIMESTAMP()	*A timestamp. TIMESTAMP values are stored as the number of seconds since the Unix epoch ('1970-01-01 00:00:00' UTC). Format: YYYY-MM-DD HH:MI:SS **Note:** The supported range is from '1970-01-01 00:00:01' UTC to '2038-01-09 03:14:07' UTC
TIME()	A time. Format: HH:MI:SS **Note:** The supported range is from '-838:59:59' to '838:59:59'
YEAR()	A year in two-digit or four-digit format.

Storage engine

- A storage engine is a software module that a database management system uses to create, read, update data from a database.
- There are two types of storage engines in MySQL: transactional and non-transactional.
- InnoDB and MyISAM
- Show engines;
  - Observe the Default, Transaction and Savepoint

Super key : Defined as set of attributes within a table that uniquely identifies each record with in a table. It is a superset of candidate keys.

Candidate key : Defined as the set of fields from which primary key can be selected. Act as a primary key for a table to uniquely identify each record in that table.

Primary key: It is a candidate key that is most appropriate to become main key of the table.

Composite key : Key that consists of two or ore attributes that uniquely identify an entity occurrence.

Secondary or Alternative key :Candidate key which are not selected for primary key

Non-key attribute : Attributes other than candidate key attributes in a table

K E Y S

A Candidate Keys is a key or combination of attributes that can be uniquely used to identify a database record without any extraneous data. Each table may have one or more Candidate Keys. One of these Candidate Keys is selected as the table Primary Key. Any of the candidate keys that is not part of the primary key is called an Alternate Key. One can describe a Candidate Key as a Super Key that contains only the minimum number of columns necessary to determine uniqueness.

Ex: The employee number and employees name are a super key because the employee number should be unique in itself, while common names like “John Smith” will be duplicated within a large database.

Primary Key

✓Column or combination of columns whose values uniquely identify each row in the table.

✓Ensures that no duplicate or null values are entered in the column (or columns) .

✓Enforces integrity of the table.

✓A table can have only one primary key constraint.

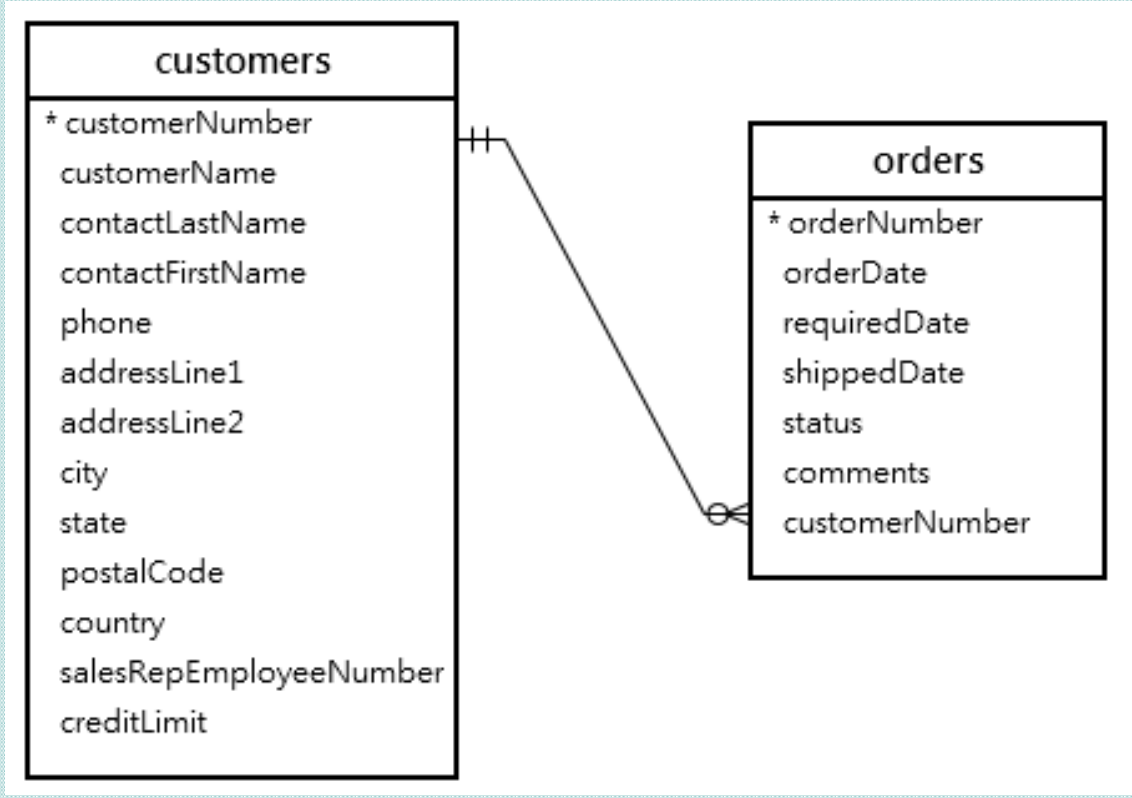
Foreign Key

✓Column or combination of columns used to establish and enforce a relationship between the data in two tables.

✓This relationship is created by adding a column(s) in one of the tables to refer to the other table's column(s) protected by PRIMARY KEY or UNIQUE constraint. This column becomes a foreign key in the first table.

Foreign Key

- is a column or group of columns in a table that links to a column or group of columns in another table.
- Customer – parent table or referenced table
- Orders is a child table or referencing table



- Referential integrity between the child and parent tables by using the ON DELETE and ON UPDATE
- Reference options are
  - Cascade - if a row from the parent table is deleted or updated, the values of the matching rows in the child table automatically deleted or updated.
  - Set NULL - if a row from the parent table is deleted or updated, the values of the foreign key column (or columns) in the child table are set to NULL.
  - Restrict - if a row from the parent table has a matching row in the child table, MySQL rejects deleting or updating rows in the parent table.

Superkey & Candidate Key Example

```
CREATE TABLE Schedule(teacher CHAR(20) not null,
Period number(3) not null check (period BETWEEN 1 and 6),
Classroom number not null,
Unique(teacher,period),                -candidate key
UNIQUE(teacher,classroom),             -candidate key
UNIQUE(period,classroom),              -candidate key
UNIQUE(teacher,period,classroom));  -super key
```

Including Constraints

What are Constraints?

- Constraints enforce rules at the table level.
- Constraints prevent the deletion of a table if there are dependencies.

The following constraint types are valid:

- NOT NULL
- UNIQUE
- PRIMARY KEY
- FOREIGN KEY
- CHECK
- DEFAULT

Integrity Constraints in CREATE TABLE

There are two ways to specify constraints

- As part of the column definition: a column constraint
- At the end of the CREATE TABLE statement: a table constraint

Nulls

- To test for nulls, only use the comparison operators: IS NULL and IS NOT NULL.
- All scalar functions return null when given a null argument.
- Most group functions ignore nulls.

Creating a Table

```
CREATE TABLE [user.]table
      ({column datatype [DEFAULT expr]
      [column_constraint | table_constraint]}
      [,{column datatype [DEFAULT expr]
      [column_constraint |
      table_constraint]}]…)[AS query ]
```

```
CREATE TABLE employees(
             employee_id  NUMBER(6),
             first_name   VARCHAR2(20),
             ...
             job_id VARCHAR2(10) NOT NULL,
             CONSTRAINT emp_emp_id_pk
              PRIMARY KEY (EMPLOYEE_ID));
```

The NOT NULL Constraint

Ensures that null values are not permitted for the column:

EMPLOYEE_ID	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	DEPARTMENT_ID
100	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	90
101	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	90
102	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	90
103	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	60
104	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	60
178	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	7000	
200	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	10

...

20 rows selected.

`NOT NULL` constraint (No row can contain a null value for this column.)

`NOT NULL` constraint

Absence of `NOT NULL` constraint (Any row can contain null for this column.)

The NOT NULL Constraint

Is defined only at the column level:

```
CREATE TABLE employees(
    employee_id     NUMBER(6),
    last_name       VARCHAR2(25) NOT NULL,
    salary          NUMBER(8,2),
    commission_pct  NUMBER(2,2),
    hire_date       DATE
                    CONSTRAINT emp_hire_date_nn
                    NOT NULL,
...
```

System named

User named

The UNIQUE Constraint

UNIQUE constraint

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	EMAIL
100	King	SKING
101	Kochhar	NKOCHHAR
102	De Haan	LDEHAAN
103	Hunold	AHUNOLD
104	Ernst	BERNST

...

INSERT INTO

208	Smith	JSMITH

Allowed

209	Smith	JSMITH

Not allowed: already exists

The UNIQUE Constraint

Defined at either the table level or the column level:

```
CREATE TABLE employees(
    employee_id       NUMBER(6),
    last_name         VARCHAR2(25) NOT NULL,
    email             VARCHAR2(25),
    salary            NUMBER(8,2),
    commission_pct    NUMBER(2,2),
    hire_date         DATE NOT NULL,
...
    CONSTRAINT emp_email_uk UNIQUE(email));
```

The PRIMARY KEY Constraint

Departments

PRIMARY KEY

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

...

INSERT INTO

Not allowed
(Null value)

	Public Accounting		1400
50	Finance	124	1500

Not allowed
(50 already exists)

The PRIMARY KEY Constraint

Defined at either the table level or the column level:

```
CREATE TABLE   departments(
    department_id         NUMBER(4),
    department_name       VARCHAR2(30)
      CONSTRAINT dept_name_nn NOT NULL,
    manager_id            NUMBER(6),
    location id           NUMBER(4),
     CONSTRAINT dept_id_pk PRIMARY KEY(department_id));
```

The FOREIGN KEY Constraint

DEPARTMENTS

PRIMARY KEY

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

...

EMPLOYEES

FOREIGN KEY

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
... 107	Lorentz	60

INSERT INTO

200	Ford	9
201	Ford	60

Not allowed (9 does not exist)

Allowed

The FOREIGN KEY Constraint

Defined at either the table level or the column level:

```
CREATE TABLE employees(
    employee_id       NUMBER(6),
    last_name         VARCHAR2(25) NOT NULL,
    email             VARCHAR2(25),
    salary            NUMBER(8,2),
    commission_pct    NUMBER(2,2),
    hire_date         DATE NOT NULL,
...
    department_id     NUMBER(4),
    CONSTRAINT emp_dept_fk FOREIGN KEY (department_id)
      REFERENCES departments(department_id),
    CONSTRAINT emp_email_uk UNIQUE(email));
```

FOREIGN KEY Constraint Keywords

- FOREIGN KEY: Defines the column in the child table at the table constraint level
- REFERENCES: Identifies the table and column in the parent table
- ON DELETE CASCADE: Deletes the dependent rows in the child table when a row in the parent table is deleted.
- ON DELETE SET NULL: Converts dependent foreign key values to null
- ON DELETE RESTRICT : restrict deleting

Example:

CREATE TABLE supplier(supplier_id numeric(10) not null, supplier_name varchar2(50) not null, contact_name varchar2(50), CONSTRAINT supplier_pk PRIMARY KEY (supplier_id));

CREATE TABLE products(product_id numeric(10) not null, supplier_id numeric(10) not null, CONSTRAINT fk_supplier FOREIGN KEY (supplier_id) REFERENCES supplier(supplier_id) ON DELETE CASCADE);

CREATE TABLE products(product_id numeric(10) not null, supplier_id numeric(10) not null, CONSTRAINT fk_supplier FOREIGN KEY (supplier_id) REFERENCES supplier(supplier_id) ON DELETE SET NULL);

The CHECK Constraint

- Defines a condition that each row must satisfy

```
create table test_chk (
   id smallint AUTO_INCREMENT,
   age tinyint not null,
   primary key(id),
   check (age<20)
);
```

The ALTER TABLE Statement

Use the ALTER TABLE statement to:

- Add a new column
- Modify an existing column
- Define a default value for the new column
- Drop a column

```
ALTER TABLE    table
ADD [CONSTRAINT constraint] type (column);
```

```
ALTER TABLE table
MODIFY      (column datatype [DEFAULT expr]
            [, column datatype]...);
```

```
ALTER TABLE table
DROP        (column);
```

Adding a Column

- You use the ADD clause to add columns.
- The new column becomes the last column.

```
ALTER TABLE dept80 ADD (job_id VARCHAR(9));

Table altered.
```

EMPLOYEE_ID	LAST_NAME	ANNSAL	HIRE_DATE	JOB_ID
149	Zlotkey	126000	29-JAN-00	
174	Abel	132000	11-MAY-96	
176	Taylor	103200	24-MAR-98	

Modifying a Column

- You can change a column's data type, size, and default value.
- A change to the default value affects only subsequent insertions to the table.

```
ALTER TABLE  dept80
MODIFY (last_name VARCHAR(30));

Table altered.
```

Dropping a Column

Use the DROP COLUMN clause to drop columns you no longer need from the table.

```
ALTER TABLE  dept80
DROP COLUMN  job_id;

Table altered.
```

Adding a Constraint

Use the ALTER TABLE statement to:

- Add or drop a constraint.
- Enable or disable constraints
- Add a NOT NULL constraint by using the MODIFY clause

```
▪ALTER TABLE emp ADD(thriftplan Double(7,2),loancode CHAR(1)
NOT NULL);

▪ALTER TABLE emp MODIFY (thriftplan DOUBLE(9,2));

▪ALTER TABLE ship_cont ADD PRIMARY KEY(ship_no, container_no)

▪ALTER TABLE phone_calls ADD CONSTRAINT fk_areaco_phoneno
FOREIGN KEY (areaco, phoneno) REFERENCES customers(areaco,
phoneno);
```

Adding a Constraint Conti…..

Add a FOREIGN KEY constraint to the EMPLOYEES table indicating that a manager must already exist as a valid employee in the EMPLOYEES table.

```
ALTER TABLE     employees
ADD CONSTRAINT  emp_manager_fk
  FOREIGN KEY(manager_id)
  REFERENCES employees(employee_id);

Table altered.
```

Dropping a Constraint

- Remove the manager constraint from the EMPLOYEES table.
- Remove the PRIMARY KEY constraint on the DEPARTMENTS table and drop the associated FOREIGN KEY constraint on the EMPLOYEES.DEPARTMENT_ID column.

```
ALTER TABLE       employees
DROP CONSTRAINT   emp_manager_fk;
Table altered.
```

```
ALTER TABLE departments
DROP PRIMARY KEY CASCADE;
Table altered.
```

Dropping a Table

- All data and structure in the table is deleted.
- Any pending transactions are committed.
- All indexes are dropped.
- You *cannot* roll back the DROP TABLE statement.

```
DROP TABLE dept80;

Table dropped.
```

Dropping a Table…….

DROP TABLE is used to remove a table and all its data from the database

Syntax:

DROP TABLE <table name>;

Example:

DROP TABLE emp;

Changing the Name of an Object

- To change the name of a table, view, sequence, or synonym, you execute the RENAME statement.
- You must be the owner of the object.

```
RENAME dept TO detail_dept;

Table renamed.
```

Truncating a Table

- The TRUNCATE TABLE statement:
  - Removes all rows from a table
  - Releases the storage space used by that table
- You cannot roll back row removal when using TRUNCATE.
- Alternatively, you can remove rows by using the DELETE statement.

```
TRUNCATE TABLE detail_dept;

Table truncated.
```