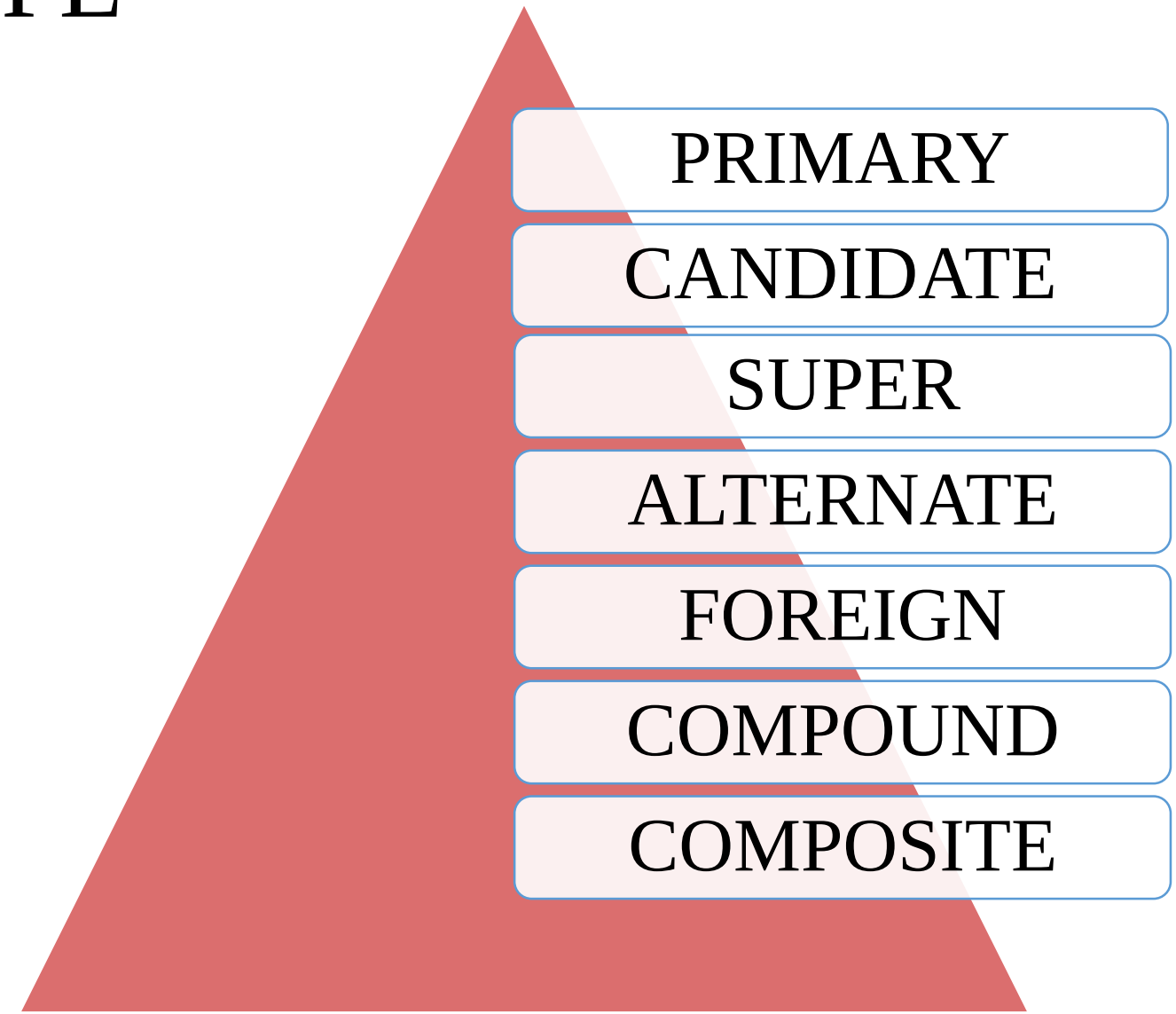
KEYS in SQL

KEYS

- Play an important role in the relational database.
- It is an attribute or set of attributes
  - which helps you to identify a row(tuple) in a relation(table).
- It allow to find the relation between two tables.
- It helps
  - uniquely identify a row in a table by a combination of one or more columns in that table.

KEY: TYPE


PRIMARY
CANDIDATE
SUPER
ALTERNATE
FOREIGN
COMPOUND
COMPOSITE

Primary Key

- A single field or combination of fields that uniquely defines a record.
- None of the fields that are part of the primary key can contain a NULL value
- A table can have only one primary key(unique).
  - Ex-Student_id or <Student_id, Student name> are primary key

Student_id	Student_name	Student_branch
coe19d001	Venketesh	CSE
coe19d002	Shree Prakash	CSE

Candidate key

- Candidate key is a primary key
- Definition:
  - Subset of primary key is not a primary key
  - Student_id is a candidate key
  - <Student_id, Student name> is not a candidate key

Student_id	Student_name	Student_branch
coe19d001	Venketesh	CSE
coe19d002	Shree Prakash	CSE

primary key in MySQL

- A primary key is created using
  - either a CREATE TABLE statement

    or
  - an ALTER TABLE statement.

- ALTER TABLE statement in MySQL
  - to drop, disable or enable a primary key.

Primary Key - Using CREATE TABLE statement

```
CREATE TABLE table_name ( column1 column_definition,
column2 column_definition, ...
CONSTRAINT [constraint_name]
PRIMARY KEY
(column1, column2, ... column_n) );
```

table_name: The name of the table that you wish to create.

column1, column2

The columns that you wish to create in the table.

constraint_name

The name of the primary key.

column1, column2, ... column_n

The columns that make up the primary key.

Example1

```
CREATE TABLE contact ( contact_id INT(11) NOT NULL,
 last_name VARCHAR(30) NOT NULL, first_name
VARCHAR(25),
CONSTRAINT contacts_pk PRIMARY KEY (contact_id) );
```

In this example, we've created a primary key on the ***contacts*** table called ***contacts_pk***. It consists of only one column - **the *contact_id* column**.

```
SQL> CREATE TABLE studentt (id number(4),
  2   name varchar2(50) not null, constraint test_pk primary key(id));

Table created.

SQL> desc studentt;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(4)
 NAME                                      NOT NULL VARCHAR2(50)

SQL> CREATE TABLE studenttt (id number(4),
  2   name varchar2(50) not null, primary key(id));

Table created.
```

```
SQL> SELECT * FROM studentt;

        ID NAME
---------- --------------------
         1 shristee
         2 heena
         3 mohit
         4 shashank
         5 avinash
```

Drop Primary Key

```
ALTER TABLE table_name DROP PRIMARY KEY;
```

```
ALTER TABLE studentt ADD CONSTRAINT
student_pk PRIMARY KEY(id,name);
```

Example:

```
ALTER TABLE studentt DROP PRIMARY KEY;
```

NOTE: We do not need to specify the name of the primary key as there can only be one on a table.

```
SQL> ALTER TABLE studentt DROP CONSTRAINT test_pk ;

Table altered.

SQL> desc studentt;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------
 ID                                                 NUMBER(4)
 NAME                                      NOT NULL VARCHAR2(50)
```

```
SQL> ALTER TABLE studentt DROP primary key;

Table altered.
```

Primary Key - Using ALTER TABLE statement

```
ALTER TABLE table_name ADD CONSTRAINT [ constraint_name ]
 PRIMARY KEY  (column1, column2,... column_n) ;
```

table_name
The name of the table to modify.
constraint_name
The name of the primary key.
column1, column2, ... column_n
The columns that make up the primary key.

```
SQL> ALTER TABLE studentt ADD CONSTRAINT student_pk PRIMARY KEY(id,name);

Table altered.

SQL> select * from studentt;

        ID NAME
---------- --------------------------------------------------
         1 shristee
         2 heena
         3 mohit
         4 shashank
         5 avinash

SQL> desc studentt;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------------
 ID                                        NOT NULL NUMBER(4)
 NAME                                      NOT NULL VARCHAR2(50)

SQL> ALTER TABLE studentt DROP primary key;
```

FOREIGN KEY

- A FOREIGN KEY is a field (or collection of fields) in one table that refers to the PRIMARY KEY in another table.
- The table containing the foreign key is called the child table
- The table containing the candidate key is called the referenced or parent table.
- The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.

Example

"Persons" table

PersonID	LastName	FirstName	Age
1	Hansen	Ola	30
2	Svendson	Tove	23
3	Pettersen	Kari	20

"Orders" table:

OrderID	OrderNumber	PersonID
1	77895	3
2	44678	3
3	22456	2
4	24562	1

"PersonID" column in the "Orders" table points to the "PersonID" column in the "Persons" table.

The "PersonID" column in the "Persons" table is the PRIMARY KEY in the "Persons" table.

The "PersonID" column in the "Orders" table is a FOREIGN KEY in the "Orders" table.

```
CREATE TABLE child_table (
...
CONSTRAINT fk_name
FOREIGN KEY(col1, col2,...)
 REFERENCES
parent_table(col1,col2)
ON DELETE [ CASCADE | SET
NULL ]
);
```

ON DELETE clause to specify consequence when the rows in the parent table are deleted.

ON DELETE CASCADE: if a row in the parent is deleted, then all the rows in the child table that reference the removed row will be deleted.

ON DELETE SET NULL: if a row in the parent is deleted, then all the rows in the child table reference the removed row will be set to NULL for the foreign key columns.

```
SQL> Create table Persons(PersonID number, LastName varchar2(20)
  2  , FirstName varchar2(20),Age number, primary key(PersonID));

Table created.

SQL> create table orders(orderID number NOT NULL,orderNumber number NOT NULL,
  2  PersonID number,PRIMARY KEY (orderID), FOREIGN KEY (PersonID)
  3  REFERENCES Persons(PersonID));

Table created.
```

```
SQL> create table orders(orderID number NOT NULL,orderNumber number NOT NULL,
  2  PersonID number, PRIMARY KEY (orderID),
  3  constraint fk_name
  4  FOREIGN KEY (PersonID)
  5  REFERENCES Persons(PersonID));

Table created.
```

```
SQL> insert into Persons values(1,'Hansen','Ola',30);

1 row created.

SQL> insert into Persons values(2,'Svendson','Tove',23);

1 row created.

SQL> insert into Persons values(3,'Pettersen','Kari',20);

1 row created.

SQL> select * from Persons;

  PERSONID LASTNAME            FIRSTNAME                 AGE
---------- ------------------- ------------------- ----------
         1 Hansen              Ola                         30
         2 Svendson            Tove                        23
         3 Pettersen           Kari                        20
```

```
SQL> insert into orders values(1,77895,3);

1 row created.

SQL> insert into orders values(2,44678,3);

1 row created.

SQL>
SQL> insert into orders values(3,24562,2);

1 row created.

SQL> select * from orders;

   ORDERID ORDERNUMBER   PERSONID
---------- ----------- ----------
         1       77895          3
         2       44678          3
         3       24562          2
```

```
SQL> insert into orders values(3,24562,4);
insert into orders values(3,24562,4)
*
ERROR at line 1:
ORA-00001: unique constraint (SCOTT.SYS_C005446) violated
```

```
SQL> delete from Persons where personID=3;
delete from Persons where personID=3
*
ERROR at line 1:
ORA-02292: integrity constraint (SCOTT.SYS_C005447) violated - child record
found
```

FOREIGN KEY on ALTER TABLE

- To create a FOREIGN KEY constraint on the "PersonID" column when the "Orders" table is already created, use the following SQL:
  - ALTER TABLE Orders
    ADD FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);
- FOREIGN KEY constraint on multiple columns, SQL syntax:
  - ALTER TABLE Orders
    ADD CONSTRAINT FK_PersonOrder
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID);

DROP a FOREIGN KEY

- ALTER TABLE Orders
  DROP FOREIGN KEY FK_name;

- Example:
  - ALTER TABLE Orders
    DROP CONSTRAINT FK_name;

- ALTER TABLE child_table DISABLE CONSTRAINT fk_name;

- ALTER TABLE child_table ENABLE CONSTRAINT fk_name;

```
SQL> drop table orders;

Table dropped.

SQL> create table orders(orderID number NOT NULL,orderNumber number NOT NULL,
  2  PersonID number, PRIMARY KEY (orderID),
  3  constraint fk_name
  4  FOREIGN KEY (PersonID)
  5  REFERENCES Persons(PersonID)
  6  on delete set null );

Table created.

SQL> insert into orders values(1,77895,3);

1 row created.

SQL>
SQL> insert into orders values(2,44678,3);

1 row created.

SQL>
SQL> insert into orders values(3,24562,2);

1 row created.

SQL> delete from Persons where PersonID=3;

1 row deleted.

SQL> select * from orders;

   ORDERID ORDERNUMBER   PERSONID
---------- ----------- ----------
         1       77895
         2       44678
         3       24562          2
```

SUPER KEY

- a superset of a candidate key.
- a set of an attribute which can uniquely identify a tuple

```
mysql> select *from contact;
+------------+-----------+------------+
| contact_id | last_name | first_name |
+------------+-----------+------------+
|          6 | shree     | prakash    |
|          7 | shree     | Ramesh     |
```

(contact_id, last_name) is a super key

COMPOSITE KEY

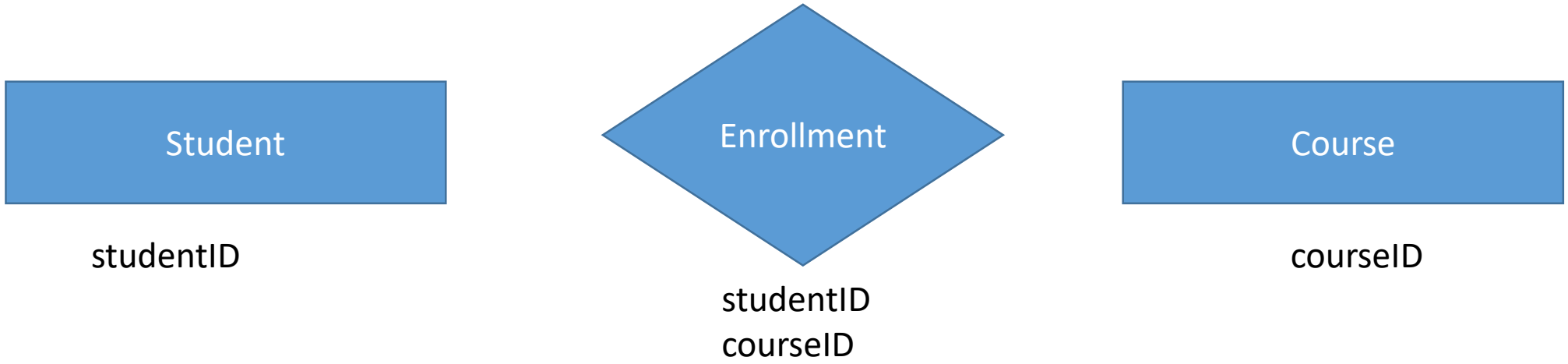
- combination of two or more columns that uniquely identify rows in a table.
- Field may or may not a primary key
- there combination must be unique

(last_name, first_name)

```
mysql> select *from contact;
+------------+-----------+------------+
| contact_id | last_name | first_name |
+------------+-----------+------------+
|          6 | shree     | prakash    |
|          7 | shree     | Ramesh     |
```

Compound key

- Compound keys are always made up of two or more primary keys from other tables.
- In their own tables, both of these keys uniquely identify data
- but in the table using the compound key they are both needed to uniquely identify data



Alternate Key/ Secondary keys

- The keys that contain all the properties needed to become a Candidate Key are known as Alternate Keys. .
- Properties
  - A Primary Key can’t be an Alternate Key. For a table with a single Candidate Key which has to be the Primary Key will not contain any Alternate Key.
  - A Foreign Key can’t be an Alternate Key as it is only used to reference another table.
  - The alternate Key should be unique.
  - An Alternate Key can be a set of a single attribute or multiple attributes.
  - It can be NULL as well

LAB EXERCISE

Ord_num	Ord_amount	Advance_amount	Ord_date	Cust_code	Agent_code	Ord_Description
004	200	3000	15-aug-2020	C004	Ac001	Masala kulcha
007	600	5000	17-sept-2020	C006	Ac003	Biriyani
008	700	100	19-feb-2019	C007	Ac005	
009	10000	600	21-march-2010	C009	Ac004	Masala dosa
010	20	600	21-april - 2012	C006	Ac006	

Table: orders

Agent_code	Agent_name	Working_area	commision	Phone_no	country
Ac001	Ramesh	Bangalore	.15	0331234567	India
Ac002	Dinesh	Bangalore	.25	0331234568	India
Ac003	Suresh	Mumbai	.35	0331234569	London
Ac004	Kamlesh	New jersey	.68	0331234564	London
Ac005	Kartik	Chennai	.73	0331234563	India

Table: Agent

1. Consider the following table Agent(**AGENT_CODE**, AGENT_NAME, WORKING_AREA ,COMMISSION , PHONE_NO, COUNTRY) and Orders(ORD_NUM, ORD_AMOUNT, ADVANCE_AMOUNT, ORD_DATE,CUST_CODE,AGENT_CODE,ORD_DESCRIPTION) where Agent_code is foreign key in table agent.
   a. Find ord_num, ord_amount, ord_date, cust_code and agent_code lives in same country or working area is same.
   b. Retrive ord_num, ord_amount, cust_code and agent_code from the table orders where the agent_code of orders table must be the same agent_code of agents table and agent_name of agents table have atleast one ‘a’ having different working_area.

LAB EXERCISE

Table: employees

employee_id	first_name	last_name	email	phone_number	hire_date	job_id	salary	manager_id	department_id
700	Hasmukh	Patel	hp@gmail.com	7003216160	15-aug-2020	Hp003	7000		90
800	Kamlesh	Paul	kp@gmail.com	7003216170	17-feb-2020	Kp 004	8000	506	90
900	Dinesh	Gandhi	dp@yahoo.com	9136278563	19-march-2101	Dg006	20000	508	80
701	Suresh	Modi	sm@dg.com	9187653294	20-april-2015	Sm009	15000		80

2. Consider the table employees(**employee_id**, first_name, last_name, email, phone_number, hire_date, job_id, salary, manager_id, department_id)
   a. Display the employee_id, manager_id, first_name and last_name of those employees who manage other employees having individual salary less than average salary of person whose last_name starts with ‘p’ .

Lab Exercise

Salesman_id	Name	City	commission
si123@06	Lakshmi	Kolkata	.5
si123@09	Ganesh	London	.6
si123@90	Dinesh	London	.3
si123@10	Joseph	Chennai	.6
si123@19	Mahesh	Chennai	.65
si123@26	Paul Adam	London	.1
si123@67	Rahul	Kolkata	.4

Table: salesman

ord_no	Purch_amt	Ord_date	Customer_id	Salesman_id
123	600	20-aug-2010	003cd	si123@19
576	750	20-feb-2018	004cd	si123@19
579	800	20-may-20120	004cd	si123@26
600	60000	20-jan-2021	006cd	si123@10
700	745	26-jan-2021	007cd	si123@09
800	860	29-jan-2019	007cd	si123@26

Table: orders

3. Consider the tables salesman(**salesman_id**, name ,city ,commission) and Orders(ord_no, purch_amt, ord_date, customer_id, salesman_id) and salesman_id is the foreign key in table orders.
   a. Display all the orders for the salesman who belongs to the same city and the individual commission of salesman is greater than the average commission of city.
   b. Delete the salesman_id from table salesman whose commisson is greater than 0.2 and set NA for the values not available in table orders.