# PRELIMINARIES

SHOW DATABASES;

USE <db name>;

SHOW TABLES;

DESCRIBE <table name>;

# CREATION

CREATE TABLE <table name> (  
 <attribute name> <attribute type> <key/constraint>,  
 ⋮  
 <other on keys>  
);

**TYPES**:

Text, application, email

Description automatically generated

**CONSTRAINTS**:

* UNIQUE
* NOT NULL
* DEFAULT <value>
* CHECK (<this column> + condition)

Or key: PRIMARY KEY [AUTO\_INCREMENT]

**KEYS:**

* Alternative form for primary key: PRIMARY KEY (<attribute1>, <attribute2>)
* foreign key: FOREIGN KEY (<attr foreign key>) REFERENCES <table name> (<attr primary key>)

# MODIFICATION

**TABLE ELIMINATION**

DROP TABLE IF EXISTS <table name>

**SCHEMA ALTERATION**

ALTER TABLE <table name>

Followed by:

|  |  |
| --- | --- |
| ADD COLUMN <attr name> <attr type> DEFAULT <default value> <constraint>; |  |
| ALTER <attr name> SET <constraint>; | does not work with varchar attributes |
| MODIFY <attribute> <attribute type> [<constraint>]; | must specify the type. Must be used (repeating the type each time) to modify the default value of a varchar |
| ADD PRIMARY KEY (<attr1>,<attr2>); |  |
| ADD CONSTRAINT <constr name>  FOREIGN KEY (<attr name>) REFERENCES <table name>(<attr name>); |  |
| DROP COLUMN <attr name> |  |

ADD CONSTRAINT FK\_prod FOREIGN KEY (prodID) REFERENCES products(productID);

Note: in ADD COLUMNS if NOT NULL, DEFAULT must be specify

**RECORD MODIFICATION**

|  |  |  |
| --- | --- | --- |
| insertion | INSERT INTO <table name> (<attr1>, <attr2>, …, <attrn>)  VALUES (<val1>, <val2>, …, <valn>),  (<val1>, <val2>, …, <valn>); | not necessary to specify values for attributes that have a DEFAULT or that van be NOT NULL |
| update | UPDATE <table>  SET <attr a> = <value>  WHERE (<attr b> = <value>); or (<attr b> LIKE '%i'); | WHERE is for condition  LIKE for strings (?):   * % for any string * \_ for any character |
| depletion | DELETE FROM <table>  WHERE (<attr b> = <value>); or (<attr b> LIKE '%i'); |  |

**RECORD SELECTION**

|  |  |
| --- | --- |
| SELECT [DISTINCT] <list of attributes/operations> [AS <rename attributes>] *or* \*  FROM <table name> [<rename table>] | SELECT attr AS a *or*  SELECT t.attr a FROM table FROM table t  DISTINCT = unique |
| WHERE(<conditions on attributes>) | LIKE, BETWEEN, IN(<element 1>, …), AND, OR |
| GROUP BY <attribute a>  HAVING <condition on attribute b> |  |
| ORDER BY <attribute> [DESC] |  |
| LIMIT <number> |  |
| ; |  |

Operations:

POWER(attr) COUNT(attr) MIN(attr) MAX(attr) COUNT(attr) AVG(attr) SUM(attr)

Note: the operations will not change the record, but only the value returned by the query

**JOINS**

|  |  |
| --- | --- |
| INNER JOIN <other table> other ON this.foreign\_key = other.primary\_key | only select keys that are present in both tables |
| LEFT JOIN … | select all from this and the corresponding from other (or NULL if no correspondence) |
| RIGHT JOIN … | select all from other and the corresponding from this (or NULL if no correspondence) |
| … UNION … | *see slides* |

# VIEWS (masks)

CREATE VIEW <view name> AS   
SELECT …

# DCL

|  |  |
| --- | --- |
| CREATE USER <user name>@<host name> IDENTIFIED BY <user pw>; | create a new user and assign them a password |
| GRANT SELECT, UPDATE ON <db\_name>.<table\_name> TO <user>@<host>; | assign privileges to users |
| REVOKE UPDATE ON <db\_name>.<table\_name> TO <user>@<host>; | remove privileges to users |