SELECT \* from film;

SELECT \* from actor;

SELECT first\_name from actor;

SELECT first\_name, last\_name from actor;

select first\_name from actor;

select \* from city;

select first\_name, last\_name, email from customer;

select \* from film;

select distinct release\_year from film;

select distinct (release\_year) from film;

select distinct (rental\_rate) from film;

select distinct (rating) from film;

select count (\*) from payment;

select \* from payment;

select count (amount) from payment;

select (amount) from payment;

select distinct (amount) from payment;

select count (distinct (amount)) from payment;

select \* from customer where first\_name = 'Jared';

select \* from film where rental\_rate > 4;

select \* from film

where rental\_rate >4 and

replacement\_cost >= 1 and film\_id > 100 and rating = 'R';

select count(\*) from film

where rental\_rate > 4 and replacement\_cost >= 1 and

film\_id > 100 and rating = 'R';

select title from film

where rental\_rate > 4 and replacement\_cost >= 1 and

film\_id > 100 and rating = 'R';

select count (title) from film

where rental\_rate > 4 and replacement\_cost >= 1 and

film\_id > 100 and rating = 'R';

select count (title) from film

where rental\_rate > 4 and replacement\_cost >= 1 and

film\_id > 100 and rating = 'R';

select count(\*) from film

where rating = 'R' or rating = 'PG-13';

select \* from film

where rating != 'R';

select email from customer

where first\_name = 'Nancy' and last\_name = 'Thomas';

select description from film

where title = 'Outlaw Hanky';

select \* from address;

select phone from address

where address = '259 Ipoh Drive';

select \* from customer

order by first\_name;

select \* from customer

order by first\_name ASC;

select \* from customer

order by first\_name desc;

select \* from customer

order by store\_id, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id desc, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id asc, first\_name desc;

select first\_name, last\_name from customer

order by store\_id asc, first\_name desc;

select email from customer

where first\_name = 'Nancy' and last\_name = 'Thomas';

select description from film

where title = 'Outlaw Hanky';

select \* from address;

select phone from address

where address = '259 Ipoh Drive';

select \* from customer

order by first\_name;

select \* from customer

order by first\_name ASC;

select \* from customer

order by first\_name desc;

select \* from customer

order by store\_id, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id desc, first\_name asc;

select store\_id, first\_name, last\_name from customer

order by store\_id asc, first\_name desc;

select first\_name, last\_name from customer

order by store\_id asc, first\_name desc;

--the five most recent payments

select \* from payment

order by payment\_date desc;

select \* from payment

where amount != 0.00

order by payment\_date desc

limit 5;

-- limit indicates the number of rows to show

-- it should be written at the end of the codes

-- we want to reward our first 10 paying customers

-- what are the customer ids of the first 10 customers who created

-- a payment?

select \* from customer;

select \* from payment;

select customer\_id from payment

order by payment\_date asc

limit 10;

-- A customer wants to quickly rent a video to watch

-- over their short lunch break

-- what are the titles of the 5 shortest (in length of runtime)

-- movies?

select title, length from film

order by length asc

limit 5;

-- If the previous customer can watch ny movie that is 50 min r less

--in run time, how many options does she have?

select count(title) from film

where length <= 50;

-- value >= low and value <= high

-- value between low and high

-- not between

--value < low or value >high

-- value not between low and high

select \* from payment

where amount between 8 and 9;

select count(\*) from payment

where amount between 8 and 9;

select \* from payment

where payment\_date between '2007-02-01' and '2007-02-15';

select \* from payment

where payment\_date between '2007-02-01' and '2007-02-14';

select \* from payment

where amount in (0.99,1.98,1.99);

select count(\*) from payment

where amount in (0.99,1.98,1.99);

select count(\*) from payment

where amount not in (0.99,1.98,1.99);

select \* from customer

where first\_name in ('John', 'jake','Julie');

-- how many transactions were greater than $5.00?

select count(\*) from payment

where amount > 5;

-- how many actors have a first name that starts with the letter P?

select count(\*) from actor

where first\_name like 'P%';

-- how many unique districts are our customers from?

select count (distinct (district)) from address;

-- retrieve the list of names for those distinct districts from

-- the previous question

select distinct (district) from address;

--how many films have a rating of R and

--replacement cost between $5 and $15?

select \* from film;

select count(\*) from film

where rating = 'R' and replacement\_cost between 5 and 15;

-- how many films have the word Truman somewhere

-- in the title?

select count(\*) from film

where title like '%Truman%'

-- how many transactions were greater than $5.00?

select count(\*) from payment

where amount > 5;

-- how many actors have a first name that starts with the letter P?

select count(\*) from actor

where first\_name like 'P%';

-- how many unique districts are our customers from?

select count (distinct (district)) from address;

-- retrieve the list of names for those distinct districts from

-- the previous question

select distinct (district) from address;

--how many films have a rating of R and

--replacement cost between $5 and $15?

select \* from film;

select count(\*) from film

where rating = 'R' and replacement\_cost between 5 and 15;

-- how many films have the word Truman somewhere

-- in the title?

select count(\*) from film

where title like '%Truman%';

select min(replacement\_cost) from film;

select max(replacement\_cost) from film;

select min(replacement\_cost), max(replacement\_cost) from film;

select avg(replacement\_cost) from film;

select round(avg(replacement\_cost),2) from film;

select sum(replacement\_cost) from film;

-- group by must appear right after a from or where statement

select customer\_id from payment

group by customer\_id

order by customer\_id;

select customer\_id, sum(amount) from payment

group by customer\_id

order by sum(amount);

-- find the customer who made the most payment

select customer\_id, sum(amount) from payment

group by customer\_id

order by sum(amount) desc;

-- find how many transaction happened with each customer

-- find who made the most number of transactions

select customer\_id, count(amount) from payment

group by customer\_id

order by count(amount) desc;

-- group by by more than one columns

select customer\_id, staff\_id, sum(amount) from payment

group by staff\_id, customer\_id;

-- it says, for example, that customer with id 1,

-- spent 60.85 with staff having id 1

select customer\_id, staff\_id, sum(amount) from payment

group by staff\_id, customer\_id

order by customer\_id;

select customer\_id, staff\_id, sum(amount) from payment

group by staff\_id, customer\_id

order by staff\_id;

select customer\_id, staff\_id, sum(amount) from payment

group by staff\_id, customer\_id

order by sum(amount);

-- filter date

select date(payment\_date) from payment;

-- how much payment was made in each date

select date(payment\_date), sum(amount) from payment

group by date(payment\_date)

order by date(payment\_date);

select date(payment\_date), sum(amount) from payment

group by date(payment\_date)

order by sum(amount);

-- how many payment did each staff memeber

-- handle and who get the bonus?

select count(date(payment\_date)), staff\_id from payment

group by staff\_id

order by staff\_id;

-- another approach

select staff\_id, count(amount) from payment

group by staff\_id;

select staff\_id, count(\*) from payment

group by staff\_id;

select \* from film;

select avg(replacement\_cost), rating from film

group by rating;

-- what are the customer ids of the top

-- 5 customers by total amount spent?

select customer\_id, sum(amount)

from payment

group by customer\_id

order by sum(amount) desc

limit 5;

-- select compant, sum(sales)

-- from finance\_table

--where company != 'Google'

-- group by company

--It is okay to use where before group by to filter result

-- we cannot use where to filter aggregate result such as

-- sum(sales)

-- using having to filter aggreagate results

-- select compant, sum(sales)

-- from finance\_table

--where company != 'Google'

-- group by company

-- having sum(sales) > 100

select customer\_id, sum(amount) from payment

group by customer\_id

having sum(amount) > 100;

select store\_id, count(customer\_id) from customer

group by store\_id;

select store\_id, count(\*) from customer

group by store\_id;

select store\_id, count(\*) from customer

group by store\_id

having count(\*) > 300;

--what customer\_ids are eligible for plantinum status?

select count(date(payment\_date)), customer\_id from payment

group by customer\_id

having count(date(payment\_date)) >= 40;

select count(\*), customer\_id from payment

group by customer\_id

having count(\*) >= 40;

--cusotmer\_ids with spents > 100 and staff\_id = 2

select customer\_id, sum(amount) from payment

where staff\_id = 2

group by customer\_id

having sum(amount) > 100;

-- retrun the customer IDs of customers

-- who have spent at least $110 with the staff

-- member who has an ID of 2?

select customer\_id, sum(amount) from payment

where staff\_id = 2

group by customer\_id

having sum(amount) >= 110;

-- how many films begin with the letter J?

select count(title) from film

where title like 'J%';

-- what customer has the highest customer ID number

-- whose name starts with an 'E' and has an address ID

-- lower than 500?

select \* from customer;

select first\_name, last\_name from customer

where first\_name like 'E%' and address\_id < 500

order by customer\_id desc

limit 1;

**-- the as operator gets executes st the very end of a**

**-- query, meaning that we cannot use the ALIAS inside WHERE**

**--operator**

**select count(amount) as num\_transactions**

**from payment;**

**select count(\*) as num\_transactions**

**from payment;**

**-- how much each customer has spent**

**select customer\_id, sum(amount)**

**from payment**

**group by customer\_id;**

**select customer\_id, sum(amount) as total\_spent**

**from payment**

**group by customer\_id;**

**-- total\_spent is assigned as the very end and we cannot**

**-- it inside the codes, for example, we cannot say**

**-- having total\_spent > 100, because the name assignment**

**-- has not yet done**

**select customer\_id, sum(amount) as total\_spent**

**from payment**

**group by customer\_id**

**having sum(amount) > 100;**

**select customer\_id, amount**

**from payment**

**where amount > 2;**

**--inner join**

**select \* from payment**

**inner join customer**

**on payment.customer\_id = customer.customer\_id;**

**-- by using inner join we are not going to see any**

**-- customer that has never created any payment (or**

**-- purchase)**

**select payment\_id, payment.customer\_id, first\_name from payment**

**inner join customer**

**on payment.customer\_id = customer.customer\_id;**

**select payment\_id, payment.customer\_id, first\_name**

**from payment**

**inner join customer**

**on payment.customer\_id = customer.customer\_id;**

**select \* from customer**

**full outer join payment**

**on customer.customer\_id = payment.customer\_id**

**where customer.customer\_id is null**

**or payment.payment\_id is null;**

**select \* from payment;**

**--what monies are not avaiable**

**select film.film\_id, title, inventory\_id, store\_id**

**from film**

**left join inventory**

**on inventory.film\_id = film.film\_id**

**where inventory.film\_id is null;**

**-- california sales tax laws have changedand we need**

**-- to alert our customers to this through email**

**-- what are the email of customers who live in California?**

**select \* from customer;**

**select \* from address;**

**select address.district, email from customer**

**full outer join address**

**on customer.address\_id = address.address\_id**

**where district = 'California';**

**-- a customer walks in and is a huge fan of the actor**

**-- 'Nick Wahlberg' and wants to know which movie he is in**

**-- get a list of all the movies 'Nick Wahlberg' has been in**

**select title from film;**

**select actor\_id from actor**

**where first\_name = 'Nick' and last\_name = 'Wahlberg';**

**-- actor\_id is 2**

**select title from film**

**full outer join film\_actor**

**on film.film\_id = film\_actor.film\_id**

**where film\_actor.actor\_id =2;**

**select title, actor.first\_name, actor.last\_name**

**from film**

**full outer join film\_actor**

**on film.film\_id = film\_actor.film\_id**

**full outer join actor**

**on film\_actor.actor\_id = actor.actor\_id**

**where actor.first\_name = 'Nick' and actor.last\_name = 'Wahlberg';**

**select now();**

**show all;**

**select timeofday();**

**select current\_time;**

**select current\_date;**

**select extract(year from payment\_date)**

**from payment;**

**select extract (year from payment\_date) as pay\_year**

**from payment;**

**select extract(month from payment\_date) as month\_pay**

**from payment;**

**select age(payment\_date) from payment;**

**select to\_char(payment\_date, 'month YYYY') from payment;**

**select to\_char(payment\_date, 'Month YYYY') from payment;**

**select to\_char(payment\_date, 'MONTH YYYY') from payment;**

**select to\_char(payment\_date,'month YY') from payment;**

**select to\_char(payment\_date, 'mon YYYY') from payment;**

**select to\_char(payment\_date,'MM YYYY') from payment;**

**select to\_char(payment\_date, 'MM/dd/YYYY') from payment;**

**select to\_char(payment\_date,'mm/dd/yyyy') from payment;**

**select distinct extract(month from payment\_date) from payment;**

**--During which months did payments occur?**

**select distinct to\_char(payment\_date,'Month') from payment;**

**-- how many payments occured on a Monday?**

**select payment\_date from payment;**

**select count(\*) from payment**

**where to\_char(payment\_date, 'day') ='monday ';**

**--select to\_char(payment\_date, 'day') from payment;**

**select count(\*) from payment**

**where extract(dow from payment\_date) = 1;**

**select round(rental\_rate/replacement\_cost,4)\*100**

**as percent\_cost from film;**

**select length(first\_name) from customer;**

**select first\_name || ' '|| last\_name from customer;**

**select first\_name || '--'||last\_name from customer;**

**select lower(left(first\_name,1)) || lower(last\_name) ||'@gmail.com' as**

**custom\_email from customer;**

**select title, rental\_rate from film**

**where rental\_rate >**

**(select avg(rental\_rate) from film);**

**select film\_id,title**

**from film**

**where film\_id in**

**(select inventory.film\_id**

**from rental**

**inner join inventory**

**on inventory.inventory\_id = rental.inventory\_id**

**where return\_date between '2005-05-29' and '2005-05-30')**

**order by film\_id;**

**-- want to find customers who have**

**-- at least one payment>11**

**select first\_name, last\_name**

**from customer as c**

**where exists**

**(select \* from payment as p**

**where p.customer\_id = c.customer\_id**

**and amount > 11)**

**select \* from cd.facilities;**

**--You want to print out a list of all of the facilities**

**--and their cost to members. How would you retrieve a**

**--list of only facility names and costs?**

**select name,membercost from cd.facilities;**

**--How can you produce a list of facilities that charge a fee to members?**

**--Expected Results should have just 5 rows:**

**select \* from cd.facilities**

**where membercost>0;**

**--How can you produce a list of facilities that**

**--charge a fee to members, and that fee is less**

**--than 1/50th of the monthly maintenance cost?**

**--Return the facid, facility name, member cost,**

**--and monthly maintenance of the facilities in question.**

**select facid,name,membercost,monthlymaintenance**

**from cd.facilities**

**where membercost >0 and**

**membercost < (monthlymaintenance/50);**

**--select membercost, monthlymaintenance/50 from cd.facilities;**

**--How can you produce a list of all facilities**

**--with the word 'Tennis' in their name?**

**select \* from cd.facilities**

**where name like '%Tennis%';**

**--How can you retrieve the details of facilities with ID**

**--1 and 5? Try to do it without using the OR operator.**

**select \* from cd.facilities**

**where facid in (1,2);**

**--How can you produce a list of members who joined**

**--after the start of September 2012? Return the memid,**

**--surname, firstname, and joindate of the members in**

**--question.**

**select memid, surname,firstname, joindate from**

**cd.members**

**where**

**joindate > '2012-09-01'**

**;**

**--How can you produce an ordered list of the first**

**--10 surnames in the members table?**

**select distinct surname from cd.members**

**order by surname**

**limit 10;**

**--You'd like to get the signup date of your last member.**

**--How can you retrieve this information?**

**select joindate from cd.members**

**order by joindate desc**

**limit 1;**

**--Produce a count of the number of facilities that**

**--have a cost to guests of 10 or more.**

**select \* from cd.facilities;**

**select count(\*) from cd.facilities**

**where guestcost>=10;**

**--Produce a list of the total number of slots booked**

**--per facility in the month of September 2012.**

**--Produce an output table consisting of facility id**

**--and slots, sorted by the number of slots.**

**select \* from cd.bookings;**

**select facid,sum(slots) as num\_slots from cd.bookings**

**where starttime between '2012-08-31' and '2012-09-30'**

**group by facid**

**order by facid;**

**--Produce a list of facilities with more than 1000**

**--slots booked. Produce an output table consisting**

**--of facility id and total slots, sorted by facility id.**

**select facid,sum(slots) as num\_slots from cd.bookings**

**group by facid**

**having sum(slots) > 1000**

**order by facid;**

**--How can you produce a list of the start times for**

**--bookings for tennis courts, for the date '2012-09-21'?**

**--Return a list of start time and facility name pairings,**

**--ordered by the time.**

**select \* from cd.bookings;**

**select cd.bookings.starttime, cd.facilities.name**

**from cd.facilities**

**inner join cd.bookings**

**on cd.bookings.facid = cd.facilities.facid**

**where cd.facilities.name like '%Tennis%'**

**and cd.bookings.starttime >= '2012-09-21'**

**and cd.bookings.starttime < '2012-09-22';**

**select cd.bookings.starttime, cd.members.surname,**

**cd.members.firstname from cd.bookings**

**inner join cd.members**

**on cd.members.memid = cd.bookings.memid**

**where cd.members.surname = 'Farrell' and**

**cd.members.firstname = 'David';**

**--select \* from cd.members;**

**create table job(**

**job\_id serial primary key,**

**job\_name varchar(200) unique not null**

**);**

**create table account\_job(**

**user\_id integer references account(user\_id),**

**job\_id integer references job(job\_id),**

**hire\_date timestamp**

**);**

**create table account(**

**user\_id serial primary key,**

**username varchar(50) unique not null,**

**password varchar(50) not null,**

**email varchar(250) unique not null,**

**create\_on timestamp not null,**

**last\_login timestamp**

**);**

**insert into account(username, password, email, create\_on)**

**values**

**('Jose', 'password', 'jose@mail.com', current\_timestamp) insert insert into job(job\_name)**

**values**

**('asronaut')**

**insert into job(job\_name)**

**values**

**('president'),('taxi driver')**

**insert into account\_job(user\_id,job\_id,hire\_date)**

**values**

**(1,1,current\_timestamp)**

**update account**

**set last\_login = create\_on**

**update account\_job**

**set hire\_date = account.create\_on**

**from account**

**where account\_job.user\_id = account.user\_id update update account**

**set last\_login = current\_timestamp**

**returning email, create\_on, last\_login**