jdbc:

java database connectivity

java code connect to your database; fire your queries from java code

ClassLoader

classforName(“className”) will load class dynamically

Drivers will have a static block for code that is needed

Driver is responsible for converting java code calls to database calls

Java code -> database

Driver : for each database, we have a different driver

Oracle | MYSQL | sql server | postgres

4 types of driver

Java type driver: ojdbc library:

Maven: if you try to add a library into your project from a repository which is other than your central repository, you need to add this repository in your pom.xml with<repositories><repository> <id> and paste url of central repository

ctrl + shift + T : box: class/interface

driver= oracle.jdbc.driver.OracleDriver

url=jdbc:oracle:thin:@mysql.cwsqa4ruuwnu.us-east-2.rds.amazonaws.com:1521:orc1

change@ to your port for local hosting

properties file stores key values

Database: CRUD: Create, Read, Update, Delete

insert data, update data, select data, delete data

**Airline Reservation Demo Project in class**

-do while for checking type of screen input options

-use if/else for return date

-using enum for type of flight - first-class,business, coach,

select distinct brandname from product;

“distinct” is used to select unique values

select \* from product where rownum <=2;

“rownum” plus condition to select first two rows

select \* from product where mfgdate is not null;

if comparing null cannot use “<> null>” have to use “is not null”

select max(price) from product;

give maximum price value from product table

//can also use “min”

select count(product\_id) from product;

gives all products, if product\_id is null then does not count it

select avg(price) from product

select sum(price) from product

select \* from product where lower( brandname) in (‘hp’,’dell’)

select\* from product where product\_id between 101 and 110

//includes 101 and 110

select price “$” from product where product\_id between 101 and 110

select concat(price, ‘$’) from price product where product\_id between 101 and 110

select count(product\_id) as number, brandname from product

group by brandname;

//”group by” is used when you need to aggregate , title

//only use what you did in select, you can do group by

select count(product\_id) as number, brandname from product

where brandname=’dell’

group by brandname;

“where” is each in every row

select count(product\_id) as number, brandname from product

group by brandname;

having brandname = ‘dell’

will filter out

can’t use “having” without “group by”

must use them together!

//display product in ascending order of name

select \* from product order by name asc ;

//display product in descending order of name

select \* from product order by name desc;

//display product in ascending order of name then id

select \* from product order by name, product\_id asc ;

//display product in ascending order of name then id

select \* from product order by name, 1 asc ;

1 refers to your first column of that table

also use “where” with condition return with least amount of records

select \* from emp, dept where dept.id=’b’

select ename, dept\_name from department\_master, emp where dep.dept\_code=11 and emp.deptno=dept.dept\_code;

SELECT column\_name(s)

FROM table1

INNER JOIN table2

on table1.column.name =table2.column\_name

SELECT ename, dept\_name

FROM emp

INNER JOIN department\_master dept

on emp.deptno = dept.dept\_code;

deserialization returns new object and readResolve()

override deserialization so it returns same object

Singleton

objectOutputStream with getInstance()

obj that is static cannot be written into a file

cloning, implements,

factory method

maven repository tomcat library into your dependency;

BasicDataSource

getConnection is a factory method and also could be instance or static

factory returns same object has to be static

if another object has to be instance